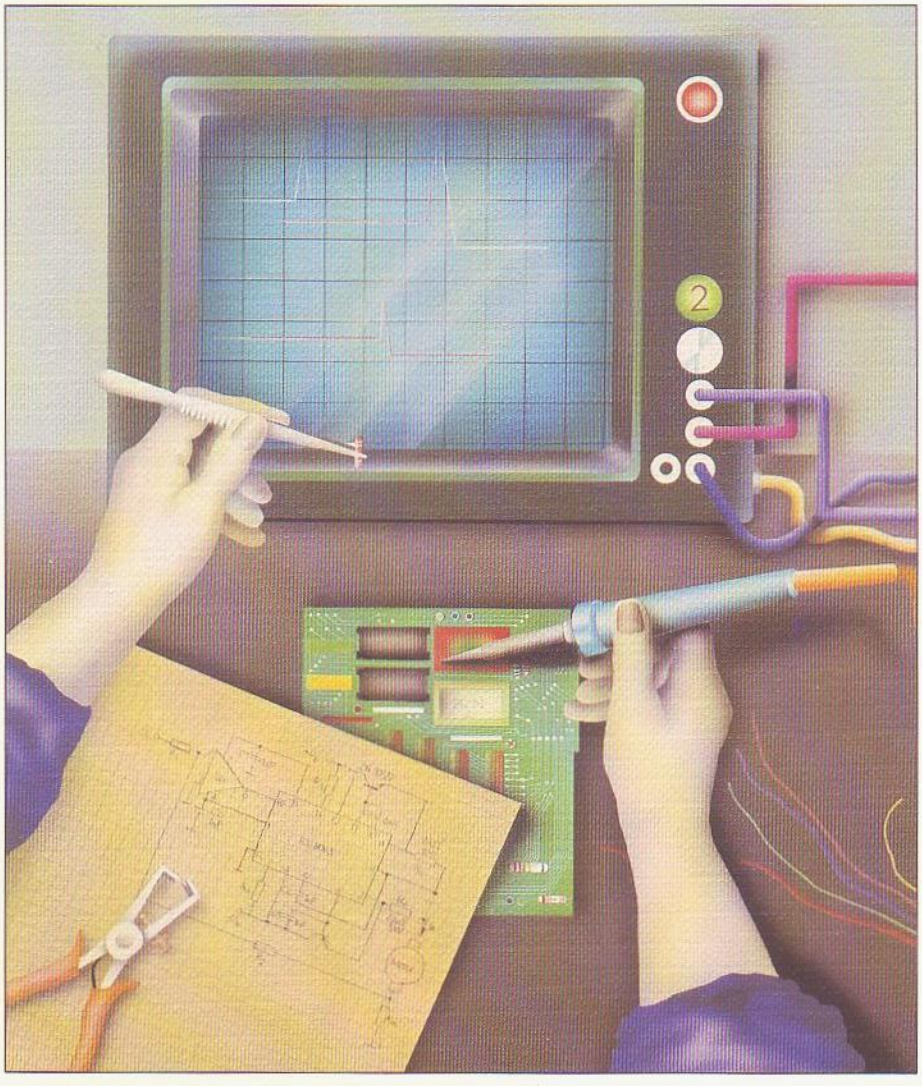
# CHIP SPECIAL 4/81

DM 24,–

Eine Sonderpublikation von CHIP. Das Mikrocomputer-Fachmagazin.

Österreich – öS 210,00; Luxemburg – lfr 430,00; Schweiz – sfr 26,00; Niederlande – hfl 30,00



# Bauanleitungen für Mikrocomputer

**Die besten Ideen, einen Computer auszubauen, zu erweitern und optimal zu nutzen.**

**Für AIM-65, Apple, CBM, TRS-80 und andere**

**Für alle bekannten Computersysteme**

# CHIP SPECIAL

## **Die umfangreichen Sonderpublikationen von CHIP** zu den aktuellsten und wichtigsten Themen aus der Welt der Mikrocomputer-Technik. Jedes Special für ganze 24 Mark!

**CHIP Special.**
**Das Plus von CHIP, das CHIP noch einmaliger macht!**

Special 1/80:
**Mikrocomputer Digest**
Das Beste von CHIP über Mikrocomputer-Grundwissen, Programmierung und Programme, Testberichte Personal-Computer und Taschenrechner, Selbstbauanleitungen, Mikrocomputer-Anwendung. Außerdem ein Computerkurs

Special 2/80:
**Programme**
für Personal-Computer und Taschenrechner, über 100 Seiten Programme mit Beschreibungen, Flußdiagrammen und Listings für Haus, Freizeit und Schule.

Special 4/80:
**Mikroprozessoren**
Eine komplette Übersicht des Mikroprozessorenangebots. Ausführliche Beschreibung von Daten, Anschlußbelegungen, Innenschaltungen etc. von Speicherbausteinen wie RAMs, ROMs und EPROMs. Dazu die gesamte Systemperipherie.

Special 1/81:
**Mikrocomputer** *
für Haus, Freizeit und Beruf. Eine umfassende Übersicht über Einplatinen-Computer, Personal-Computer, Taschenrechner und Drucker der wichtigsten zur Zeit angebotenen Geräte.

In Vorbereitung:
Special 4/81:
**Selbstbau**
Alles über Selbstbau. Eine Fülle von Bauanleitungen für Taschenrechner, Kleincomputer und Mikroprozessor-Systeme bringt dieses Special. Anfänger wie Selbstbauprofis kommen hier voll auf Ihre Kosten.

Erscheinungstermin:
November 1981

*Special 3/80 vergriffen

Die CHIP-Specials gibt es beim Zeitschriftenhandel. Oder direkt bestellen beim CHIP-Leserservice, Vogel-Verlag, D-8700 Würzburg 1, Postfach 67 40.

# CHIP SPECIAL

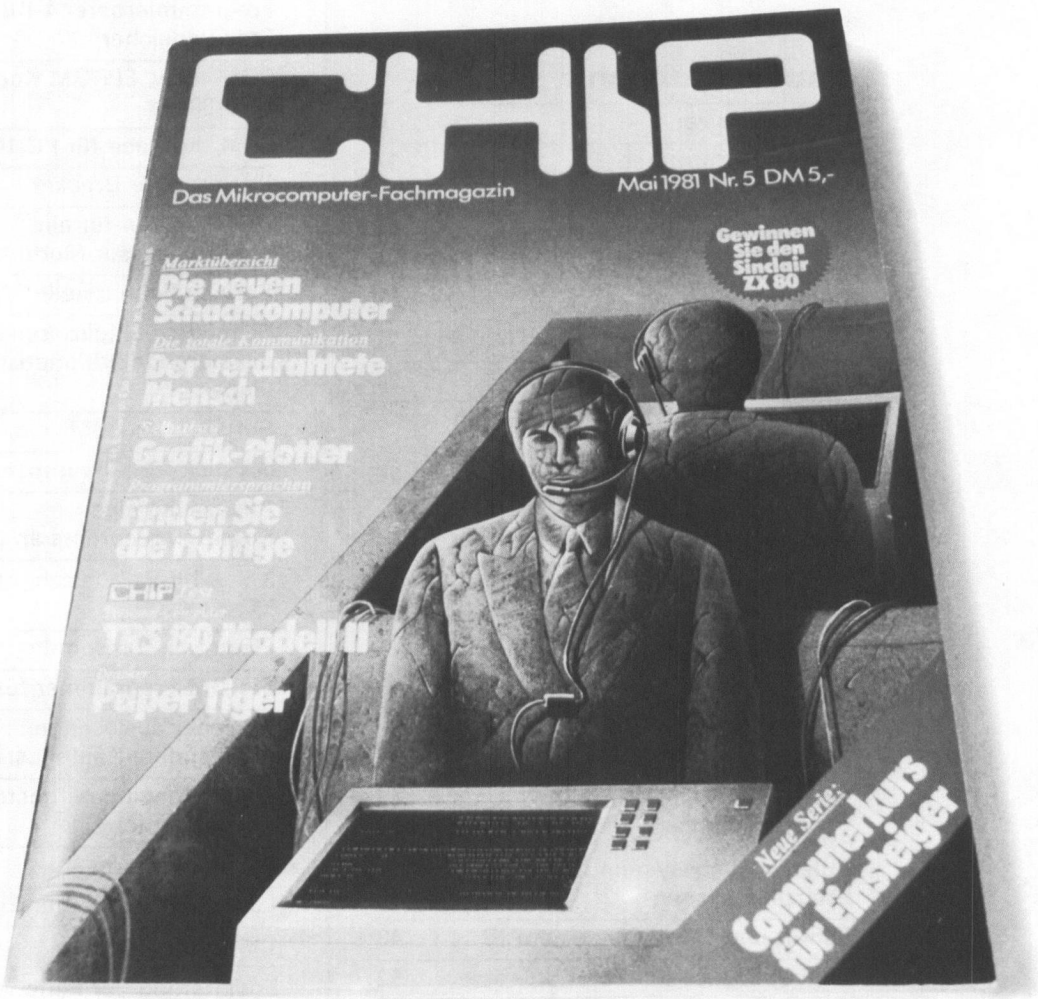
Eine Sonderpublikation von CHIP. Das Mikrocomputer-Fachmagazin.


| | |
|---|---|
| Kolumne | 5 |

## Bauanleitungen

| | |
|---|---|
| µC-Synthesizer | 6 |
| Farbgrafik-Interface | 11 |
| Interface für Apple II/ITT 2020 und Christiani-Metallpapierdrucker | 14 |
| Hochauflösende Grafik für Z-80-Computersysteme | 17 |
| Grafikerweiterung | 22 |
| Miniport für TRS-80 | 24 |
| Preiswertes Interface zur seriellen Datenausgabe für alle 6502-Mikrocomputersysteme | 26 |
| AIM-65 als interrupt-gesteuerter Frequenzzähler | 30 |
| Software-unabhängiger EPROM-Programmer | 32 |
| EPROM-Programmer für Apple II/ITT 2020 | 35 |
| Systemerweiterung für KIM1 | 38 |
| KIM1 steuert SR51 | 41 |
| Preiswerter Datendrucker | 43 |
| Analogdaten-Aufzeichnungssystem für Langzeitmessung | 45 |
| Schrift und Grafik aus einem IC | 48 |
| Analog-Digital-Umsetzer | 52 |
| Cell-Grafik für CBM | 54 |
| Uhr/Timer/Kalender für µP-Systeme | 58 |
| Computeruhr | 62 |
| Homecomputer mit Eurocom II | 65 |
| A/D- und D/A-Wandler für Sprachverarbeitung | 68 |
| Hardware-Zusatz für Computerspiele | 70 |
| Z-80-Interface für MZ-80 K | 72 |
| 220-V-Interface für Apple II | 75 |
| Tastenentprellung und Autorepeat für Video-Genie-System | 77 |
| EPROM-Programmiergerät für Z 80 | 80 |
| EPROM-Programmierboard | 82 |
| Programmierbarer 4-KB-Stapelspeicher | 84 |
| ROM- bzw. EPROM-Karte für 8080/8085 | 86 |
| CBM-Interface für PC-100-Drucker | 89 |
| Interface für Drucker | 92 |
| HEX-Tastatur für ein Mikroprozessor-Modulsystem | 95 |
| Serielle Schnittstelle für 8080/8085 | 97 |
| Schnelle multiplizierende und dividierende arithmetisch-logische Einheit | 99 |
| Softkeys für CBM | 106 |
| Joystick für Mikroprozessoren | 108 |
| Anpassung eines Kassettenrecorders an einen CBM | 113 |
| 3 × 8-bit-Parallelschnittstelle für MZ-80 K | 114 |
| Elektronik für Roboter | 116 |
| EPROM-Programmer für SDK85 | 120 |
| Speicher auslesen mit druckendem Kaufhausrechner | 123 |
| Ein mikrocomputergesteuerter Selbstbauplotter | 125 |
| Akustik-Koppler | 130 |
| Low-cost-Drucker für EUROCOM 1 | 132 |
| CBM-Programmer | 135 |
| Erweiterung der Peripherie des Tandy TRS-80-Pocket-Computers | 138 |
| Ein Experimentierbus für EUROCOM 1 | 140 |
| Hardwaretips für EUROCOM 1 | 143 |
| North-Star-BASIC im ASCII-Format | 144 |
| Dynamische Schreib-Lese-Speicher für 6502-Systeme | 148 |
| Fernschreiber T-100 am Apple II | 150 |
| Videoerweiterung für den NASCOM-1 | 152 |
| Buchservice | 110 |
| Impressum | 154 |
| Inserentenverzeichnis | 154 |

CHIP, Zeitschrift für Mikrocomputer-Technik.
Spiegelbild der ganzen Mikrocomputer-Welt.
CHIP bringt Ausgabe für Ausgabe
alles, worauf es ankommt. Für alle, die
mehr wissen wollen und mehr wissen müssen,
wenn es um Mikrocomputer-Technik geht.




CHIP gibt es
beim Zeitschriftenhandel.
Oder direkt bestellen
beim CHIP-Leserservice.

Vogel-Verlag, D-8700 Würzburg 1, Postfach 67 40.

# Zu diesem Heft

Dieses CHIP-Special entstand durch die rege, aktive und kreative Arbeit mit unseren Lesern, die sich besonders mit der Hardware der Mikroprozessoren und der Mikrocomputersysteme beschäftigen. Deshalb steht in diesem Sonderheft weniger die Software, also die Programme und die Programmierung der einzelnen Probleme, im Vordergrund, sondern das Arbeiten mit dem Lötkolben, mit aktiven und passiven Bauelementen, mit Platinen oder einer Wire-Wrap-Technik und mit integrierten Schaltkreisen.
Wesentlicher Bestandteil dieses Heftes sind die Bauanleitungen des Hardware-Wettbewerbes, den die CHIP-Redaktion im Frühjahr 1981 veranstaltet hatte.
Eigentlich wollten wir die zwei besten Bauanleitungen für unsere Leser suchen und finden. Als Themenkreise gab es in der Ausschreibung zwei Sektionen:

- Hardware für Mikroprozessoren bzw. für Mikrocomputer-systeme und
- Hardware für Taschenrechner.

Für die beiden ersten Preise wurde ein Geldbetrag von je 3000 DM ausgesetzt, insgesamt gab es über 10 000 DM zu gewinnen.
Die Teilnahme beschränkte sich nicht nur auf den Kreis der Hobbyelektroniker – der jüngste Teilnehmer war ein 12jähriger Schüler –, sondern auch auf „Profis".
Trotzdem: Die Ideen von manchen Amateuren sind absolute Spitzenleistung. Viele der professionellen Bauanleitungen zeugten von hoher Qualität.
Jeder Teilnehmer gab sein Bestes, und die Auswertung wurde für die CHIP-Redakteure ein angenehmes Problem. Es wurde viel in der Redaktion diskutiert, viel gelötet, abgestimmt und ausprobiert. Über manchen Beitrag haben wir gestaunt, über manche aber auch gelacht, aber nur wegen seiner Originalität.
Der große Durchschnitt im Niveau lag weit über unseren Erwartungen.
In diesem Heft finden Sie nicht nur Bauanleitungen und Programme, sondern auch Ideen für die Erweiterung, Ergänzung und andere Anwendungsbereiche für die Mikroprozessoren und Mikrocomputersysteme.
Die CHIP-Leser haben mit viel Mühe, erheblichem Zeitaufwand und mit großem Durchblick zahlreiche Bauanleitungen geschaffen.
Alle Arbeiten wurden fotografiert und dokumentiert, damit Sie sich auch von den technischen Möglichkeiten überzeugen können. Der Nachbau der einzelnen Anleitungen dürfte durch die zahlreichen Schaltbilder und Fotos keine großen Schwierigkeiten bereiten.
Falls die eine oder andere Bauanleitung nicht auf Anhieb funktioniert, denken Sie auch bitte daran, daß elektronische Bauteile erhebliche Toleranzen haben. Nach einem genauen Abgleich oder einer Fehlersuche bereitet Ihnen der Selbstbau viel Spaß. Der Umbau für Ihr System ist in den meisten Fällen auch möglich.
Und nun viel Spaß bei der Lektüre. Ihre CHIP-Redaktion.

Per Herrmann

# µC-Synthesizer

Nach wie vor erfreuen sich Schaltungen zur elektronischen Tonsynthese großer Beliebtheit, und mit Aufkommen der Mikroprozessoren und Homecomputer wurden auch etliche „programmierbare" Tongeneratoren veröffentlicht, vom einfachen Emitterfolger mit Lautsprecher bis zu umfangreichen Schaltungen z.B. mit dem IC 50240 (Top-Octave-Synthesizer) oder SN 76489 (ähnlich SN 76477) bzw. AY-3-8910.

Doch selbst die zuletztgenannten Tonerzeuger eignen sich eher zum Vertonen von Spielprogrammen als zur Erzeugung ansprechender „Orgel"-Musik, hauptsächlich, da die Hüllkurven vorgegeben sind und nur Rechteckschwingungen produziert werden, was die Klangformung stark einschränkt.

Die folgende Schaltung bietet die Möglichkeit, mit einem 8-bit-Prozessor in Frequenz, Amplitude und Schwingungsform freiprogrammierbare Töne zu erzeugen. Sie ist weiterhin so angelegt, daß ein Prozessor beliebig viele (natürlich innerhalb gewisser technischer Grenzen) solcher Schaltungseinheiten bedienen kann, so daß polyphones Spiel nicht nur möglich ist, sondern sogar mit Heimorgeln durchaus konkurrieren kann.

## Das Blockschaltbild

Um eine beliebige Schwingungsform aufbauen zu können, verwende ich ein 256 × 8-bit-RAM, in dem eine Schwingungsperiode digitalisiert abgespeichert wird. Sie kann dann über einen D/A-Wandler in entsprechende Spannungswerte umgesetzt werden. Um diese Schwingung mit beliebiger Frequenz $f$ abrufen zu können, werden die RAM-Adressen fortlaufend über einen 8-bit-Zähler adressiert, dieser wiederum erhält seine Taktfrequenz von einem programmierbaren Teiler, der eine Festfrequenz $F$ ganzzahlig durch $n$ teilt.

Daraus folgt:

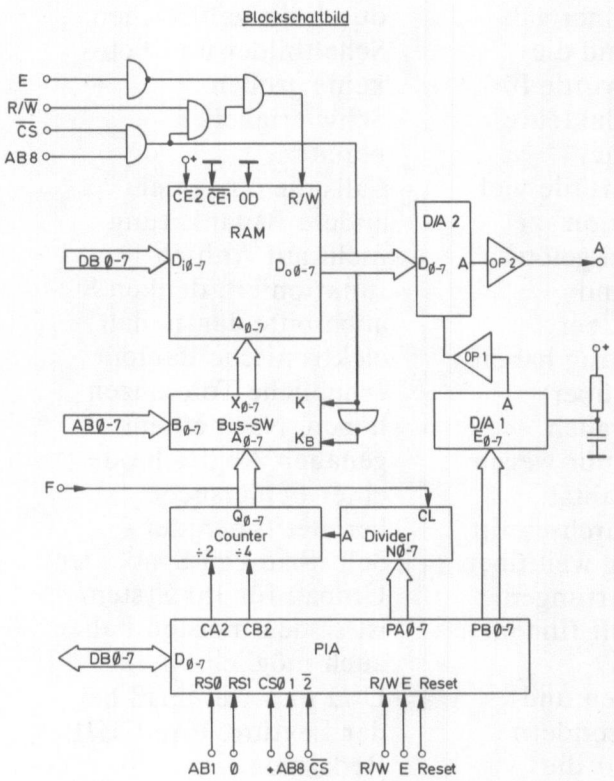
$$f = \frac{1}{256} \cdot \frac{F}{n}$$

mit $0 < n < 255$ (8-bit)

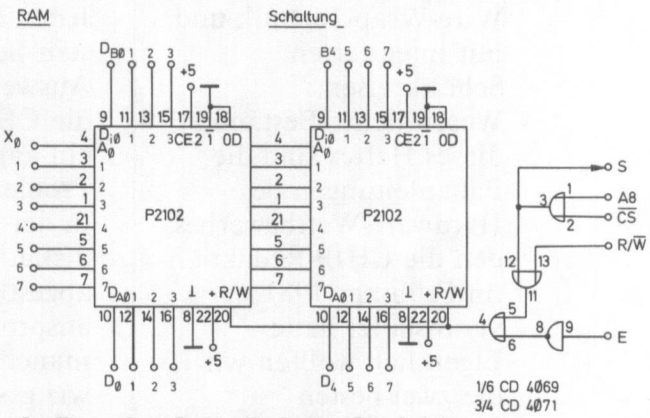
Somit würde man für eine Schwingungsfrequenz von 880 Hz (a′) bereits eine Taktfrequenz $F$ von 28,8 MHz benötigen, wenn man für $n$ einen mittleren Wert von 128 setzt, weiterhin wird die Auflösung (d.h. die „Tonschrittweite") für kleinere $n$ immer schlechter, was aufgrund des kleinen Bereiches von $n$ schnell stört.

Daher ist der Zähler so aufgebaut, daß er über zwei Kontrolleingänge (÷2, ÷4) gesteuert, nur jede zweite bzw. vierte RAM-Adresse anwählen kann, womit einfache Oktavsprünge ohne Änderung von $n$ möglich sind. Dabei wird die RAM-Auflösung herabgesetzt, die aber mit 256 Werten ohnehin recht hoch ist.

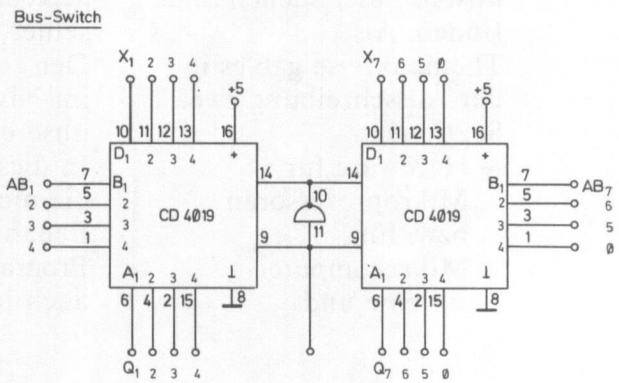
Schließlich will man noch die Amplitude des Ausgangssignals beliebig regeln können (Hüllkurve), dazu steuere ich einfach die Referenzspannung (aus der der D/A-Wandler seine Ausgangsspannung ableitet) von D/A 2 über einen zweiten 8 bit-D/A-Wandler (D/A1). Nach genauem Offset-Abgleich des Puffers OP1 läßt sich die Ausgangsspannung an A




**Blockschaltbild des µC-Synthesizer**

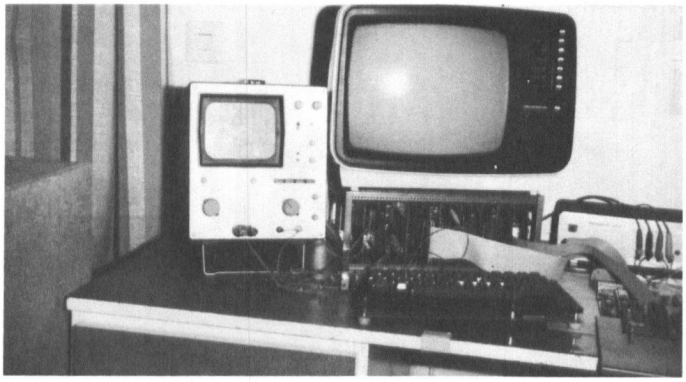



**Aufbau des Schreib-Lese-Speichers**




**Bausteine für den Bus-Schalter-Betrieb**

**Aufbau für die Erprobung der Schaltung des µP-Synthesizer**



**Ansicht des Rechners mit dem 6802 beim Autor**

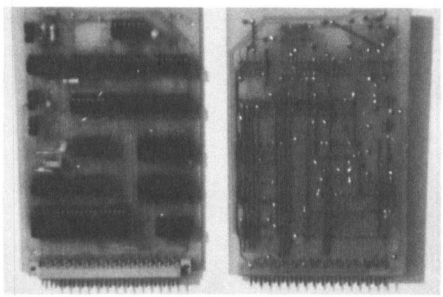
```
                +++ SINUSTABELLE +++

      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0000 - 82 86 89 8C 8F 92 95 98 9B 9E A1 A4 A7 AA AD B0
0010 - B3 B6 B9 BB BE C1 C4 C6 C9 CB CE D0 D3 D5 D7 DA
0020 - DC DE E0 E2 E4 E6 E8 E9 EB ED EE F0 F1 F3 F4 F5
0030 - F6 F8 F9 F9 FA FB FC FD FD FE FE FE FF FF FF FF
0040 - FF FF FF FE FE FE FD FD FC FB FA F9 F9 F8 F6 F5
0050 - F4 F3 F1 F0 EE ED EB E9 E8 E6 E4 E2 E0 DE DC DA
0060 - D7 D5 D3 D0 CE CB C9 C6 C4 C1 BE BB B9 B6 B3 B0
0070 - AD AA A7 A4 A1 9E 9B 98 95 92 8F 8C 89 86 82 7F
0080 - 7C 79 76 73 70 6D 69 66 63 60 5D 5A 57 54 51 4E
0090 - 4B 49 46 43 40 3E 3B 38 36 33 31 2E 2C 29 27 25
00A0 - 23 21 1F 1C 1B 19 17 15 13 12 10 0F 0D 0C 0A 09
00B0 - 08 07 06 05 04 03 03 02 01 01 00 00 00 00 00 00
00C0 - 00 00 00 00 00 01 01 02 03 03 04 05 06 07 08 09
00D0 - 0A 0C 0D 0F 10 12 13 15 17 19 1B 1C 1F 21 23 25
00E0 - 27 29 2C 2E 31 33 36 38 3B 3E 40 43 46 49 4B 4E
00F0 - 51 54 57 5A 5D 60 63 66 69 6D 70 73 76 79 7C 7F
```

**Annäherung einer Schwingung (Sinus)**
Nach Abspeichern dieser 256 Werte in den Schwingungsspeicher (RAM) erzeugt die Oszillatorplatine eine exakte Sinusschwingung. Durch Ändern von Amplitude (ganzzahlige Division aller Werte) und Frequenz (nur jeden 2., 3. usw. Wert verwenden) können die einzelnen Teilschwingungen einer Fourierreihe leicht erzeugt werden. Addiert man diese noch, so kann man mit der Sinustabelle bel. periodische Schwingungen erzeugen.
Bsp.: Annäherung einer Rechteckschwingung:
$F(t) = \sin(\omega t) + {}^1/_3 \sin(3\,\omega t) + {}^1/_5 \sin(5\,\omega t) + {}^1/_7 \sin(7\,\omega t) \ldots$ usw.
Ampl. durch 3 div. ↑ 3fache Frequenz
Ergebnis siehe nächste Seite (für die ersten 5 Glieder) (Teilschwingungen → schwarz, Zwischensummen → blau, Ergebnis → rot)

vollständig unterdrücken, so daß sich der Oszillator „ausschalten" läßt (für Amplitude = 0).

## Die Schaltung

Das RAM setzt sich aus zwei 2102 zusammen, sie haben den Vorteil getrennter Datenein- und Ausgänge, wodurch ich einen Tri-State-Schalter sparen kann.
Der Zähler besteht aus einem 6-bit-Zähler CD4024, die beiden niedrigsten Adressen werden über zwei Flipflops (CD4027) erzeugt, dabei läßt sich Q0 (bzw. Q1) konstant auf L-Pegel setzen, wenn ÷2 (bzw. ÷4) auf H-Pegel geht. Gleichzeitig wird die Eingangsfrequenz auf die nächste Zählerstufe durchgeschaltet.
Zwei CD4526 lassen sich einfach zu einem programmierbaren Teiler durch 8 bit zusammensetzen, und die beiden D/A-Wandler sind ebenfalls als komplette Bausteine ausgeführt, hier die Typen ZN426. Sie sind mit jeweils einem Operationsverstärker gepuffert, dabei verstärkt OP1 einfach und OP2 ca. 1,7fach, beide sind in einem LM747.
Außer der PIA, die wie oben beschrieben den Teiler, D/A 1 und die Steuerleitungen ÷2, ÷4 steuert, sind noch ODER- (4071) und NICHT- (4069) Gatter nötig, die den DMA ermöglichen.
So erreichen die CD4526 bereits bei etwa 2 MHz ihre Grenzfrequenz, was den Gesamtfrequenzbereich nach oben stark einschränkt. Hier wären zwei TTL-Teiler 7497 sinnvoll, sie kosten allerdings das Dreifache, andererseits ermöglichen sie

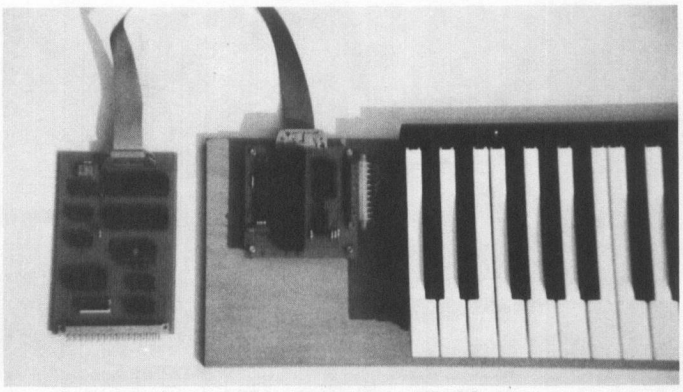


**Vorder- und Rückansicht der Platine für den Synthesizer**

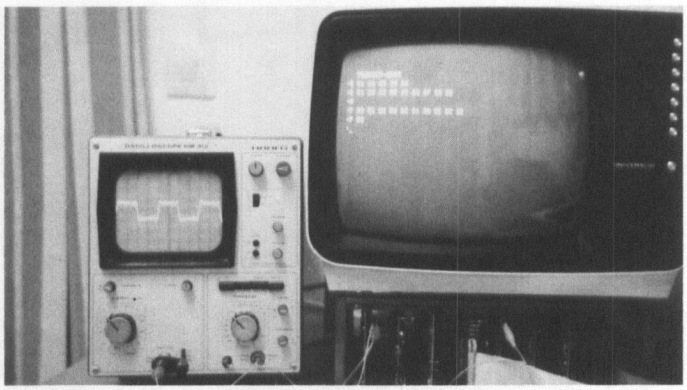
dann eine 12-bit-Teilung, was bei PIA mit 3 × 8-bit (80XX) oder 16-bit-Systemen interessant ist.

## Das Gesamtsystem

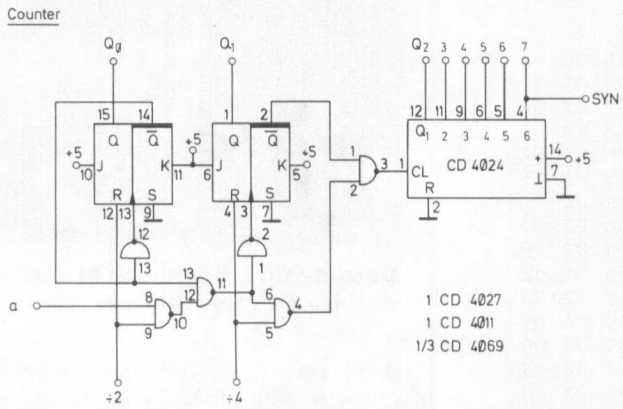
Eine „Oszillator-Platine" verhält sich aus der Sicht des Prozessors wie ein RAM, dabei wird 0,25K vom Schwingungsspeicher und 0,25K von der PIA belegt.
Somit lassen sich auf einfachste Weise mehrere solcher Platinen parallel schalten, nur der als Steckstift ausgeführte CS-Anschluß muß auf jeweils unterschiedliche Adressen gelegt werden.
Alle Platinen geben ihr Ausgangssignal A auf eine gemeinsame NF-Leitung (der 10K-Ausgangswiderstand ermöglicht die Addition der Schwingungen), und eine gemeinsame Taktfrequenzleitung versorgt sie mit der Grundfrequenz *F*.
Man benötigt also noch einen Taktgenerator, der als Quarzoszillator ausgeführt werden kann (hohe Frequenzkonstanz). Am besten baut man ihn auf einer zusätzlichen I/O-Platine auf, hier kann auch noch ein kleiner Nf-Verstärker (z.B. TDA 2002) Platz finden. Um mit dem System eine komplette Orgel „programmieren" zu können, wird über eine auf der I/O-Platine befindliche PIA eine Orgel-Klaviatur (vier Oktaven) angeschlossen, sie kann im Multiplexverfahren abgefragt werden. Zwei ebenfalls vorgesehene EPROM können Festprogramme enthalten, so daß für ein lauffähiges Minimalsystem nur noch eine CPU-Platine benötigt wird.



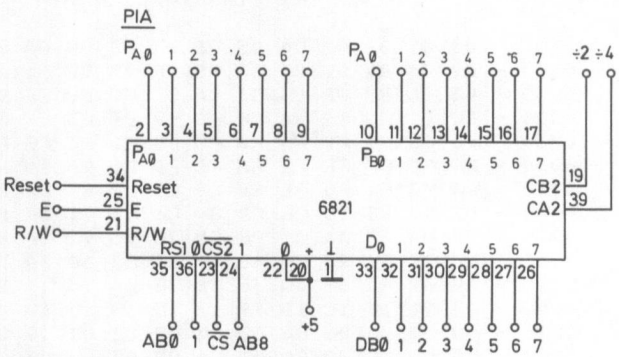
**Synthesizer an der Tastatur des Klaviers**



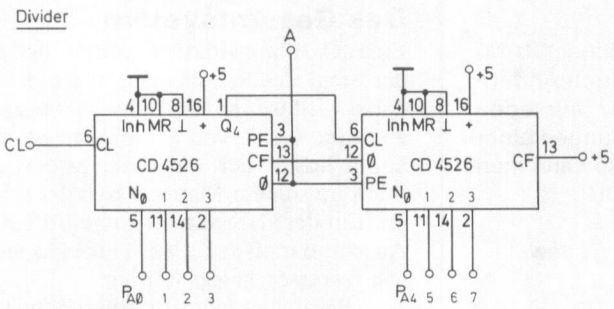
**Programmablauf mit Rechteckgenerator**



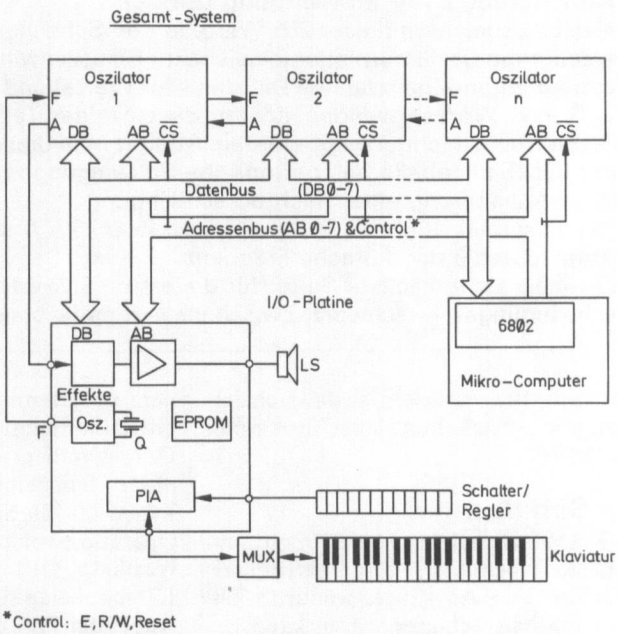

**Schaltung des programmierbaren Teilers**



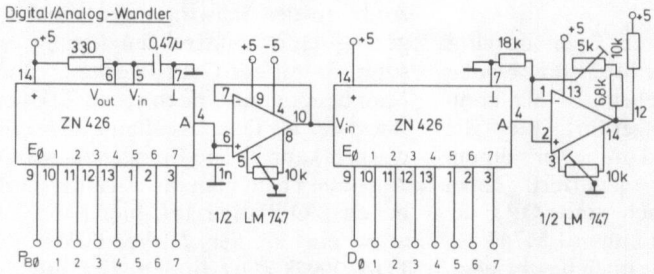

**Aufbau und Beschaltung der PIA**




**Aufbau des Zusatzteilers**




**Blockschaltbild für das gesamte System**




**Schaltung des Digital-Analog-Wandlers**

## Die Software

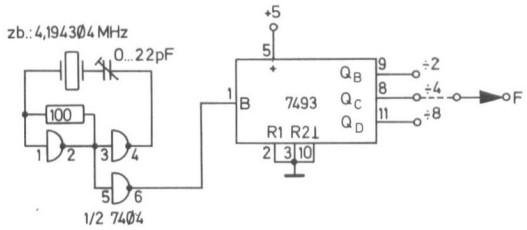
Um nun mit einer angeschlossenen Platine einen Ton zu erzeugen, muß zuerst eine aus 256 Werten bestehende Schwingungsform in den Schwingungsspeicher geladen werden. Darauf kann man den Teilfaktor für die gewünschte Frequenz in den PIA-Port A und einen der Lautstärke entsprechend proportionalen Wert in B speichern.

Alle Werte sind 8-bit-Werte, d.h., sie können zwischen 0 und 255 liegen, der Teilfaktor 0 ist allerdings nicht zulässig.

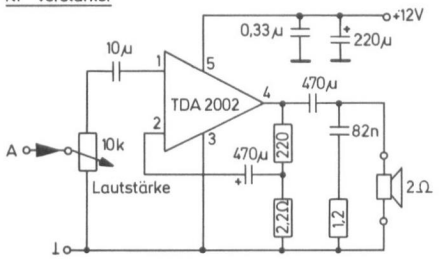
Soll sich nun die Frequenz oder die Amplitude ändern, so müssen natürlich die entsprechenden Werte neu berechnet und gespeichert werden, eine komplette Änderung der Schwingungsform, z.B. um Einschwingvorgänge nachzuahmen, benötigt viel Rechenzeit und wird daher in der Praxis nur selten vorkommen.

Schaltungsvorschläge: I/O-Platine(n)

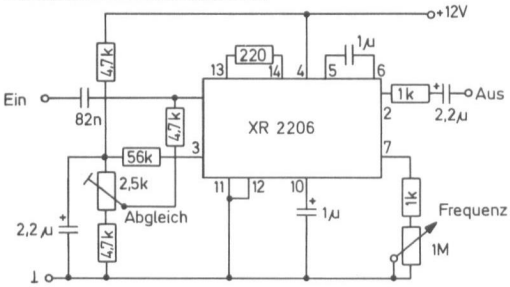
Taktgenerator




NF-Verstärker

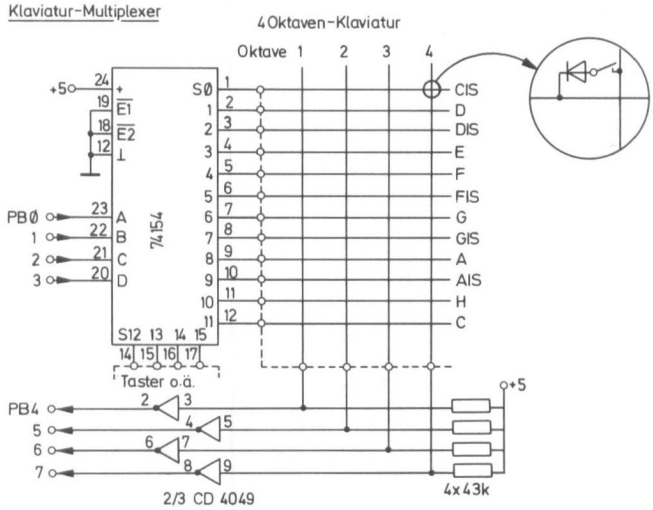



Tremolo-Effekt mit Ringmodulator



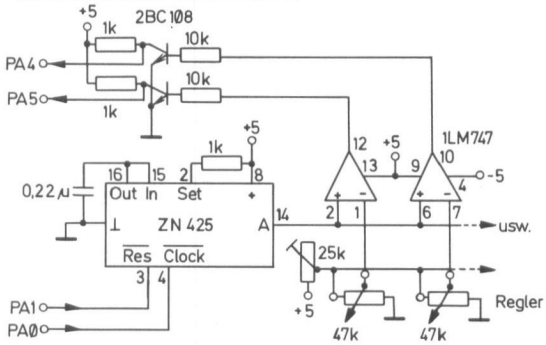

**Schaltungen für den Taktgenerator, für den Nf-Verstärker und zur Erzeugung des Tremolo-Effektes**

Klaviatur-Multiplexer

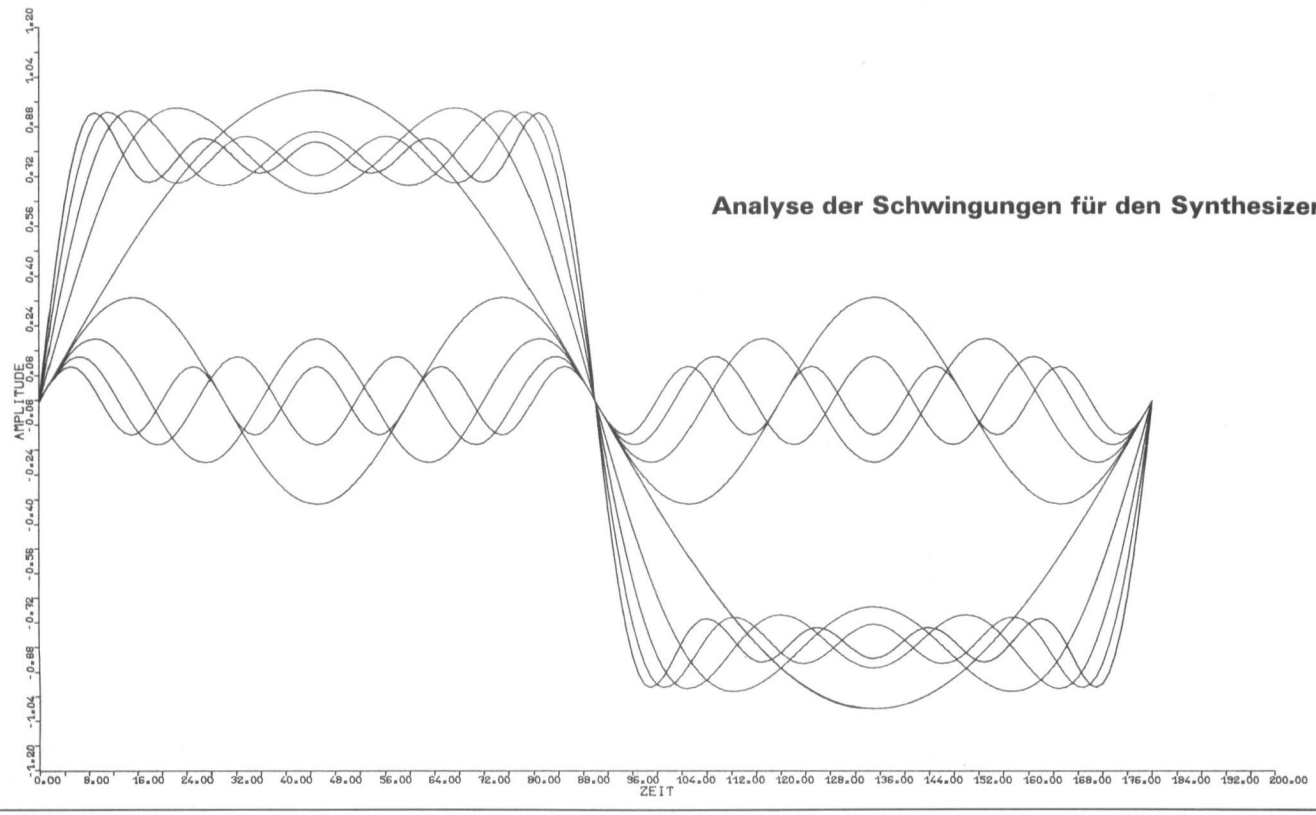



**Aufbau für die Tastatur des Klaviers**

A/D-Wandler für Analog-Eingaben




**Analog-Digital-Wandler für den Synthesizer**




**Analyse der Schwingungen für den Synthesizer**

(Falls trotzdem eine andauernde Klangänderung gewünscht ist, bietet es sich an, nur einzelne Punkte der Schwingungskurve zu verändern, hier kann viel experimentiert werden.)
So besehen, erscheint die benötigte Software etwas umfangreich, tatsächlich ist die Programmierung aber nicht wesentlich aufwendiger als die gezeigten Schaltungen, vorausgesetzt, man ändert die Schwingungsform nicht während des Spiels.
Eine interessante Möglichkeit bietet sich bei der Erzeugung der Schwingungsformen: Statt alle 256 Werte der Kurve einzeln zu erstellen, baut man sich über ein Hilfsprogramm die Schwingung mittels einer Fourier-Synthese auf:
Nach Fourier läßt sich jede periodische Schwingung als die Summe von Sinusschwingungen mit ganzzahlig vielfacher Frequenz bei geeigneten Amplituden $A_n$ der einzelnen Schwingungen erzeugen:

$$F(+) = A_0 + \sum_{n=1}^{\infty} A_n \cdot \sin \cdot n(\omega t + \varphi_n),\ n \varepsilon N^+$$

Dabei ist die Phasenverschiebung $\varphi_n$ nicht klangbestimmend. Sie kann also vernachlässigt werden, ebenso natürlich $A_0$.
Diese Darstellung einer Schwingung durch ihre sog. Harmonischen ermöglicht auf einfache Art Klangexperimente, insbesondere lassen sich Spektralanalysen von Instrumenten direkt verwenden. Außerdem läßt sich so viel Speicherplatz einsparen, nur noch eine Sinusschwingung muß komplett abgespeichert werden.
Möchte man (z.B. bei Orgel-Programmen) die Frequenzen der Tonleiter erzeugen, so berechnet man einfach eine entsprechende Tabelle der günstigsten Teilfaktoren für die gewünschten Frequenzen.
Die Erzeugung der Hüllkurven H(t) läßt sich am einfachsten über kurze Berechnungen anstellen:
H(t+1) = H(t) +/− C
für linearen Anstieg/Abfall
H(t+1) = H(t) */− C
für exponentiellen Anstieg/Abfall mit C = Konstante
Die Programmierung einer polyphon arbeitenden Orgel stellt zugegebenermaßen gewisse Anforderungen, aber das Ergebnis lohnt die Mühe.

## Wichtige Hinweise

Die freie Programmierbarkeit der Tongeneratoren eröffnet viele (Effekt-)Möglichkeiten, die sonst nur bei sehr teuren und aufwendigen „analogen" Orgeln angeboten werden.

Teilfaktoren für Tonleiter (harmonische Stimmung)

QUARZFREQUENZ : 2097152 *(F)

| SOLLFREQ. | ISTFREQ. | ABW. | PROZENT | TEILFAKTOR |
|---|---|---|---|---|
| 27.5000 | 27.4899 | .0101 | .0366 | 298 - 012A |
| 29.1352 | 29.1530 | -.0178 | .0611 | 281 - 0119 |
| 30.8677 | 30.9132 | -.0455 | .1474 | 265 - 0109 |
| 32.7032 | 32.7680 | -.0648 | .1982 | 250 - 00FA |
| 34.6478 | 34.7119 | -.0640 | .1848 | 236 - 00EC |
| 36.7081 | 36.7354 | -.0273 | .0745 | 223 - 00DF |
| 38.8909 | 38.8246 | .0662 | .1703 | 211 - 00D3 |
| 41.2034 | 41.1658 | .0376 | .0913 | 199 - 00C7 |
| 43.6535 | 43.5745 | .0791 | .1811 | 188 - 00BC |
| 46.2493 | 46.2825 | -.0332 | .0717 | 177 - 00B1 |
| 48.9994 | 49.0539 | -.0545 | .1111 | 167 - 00A7 |
| 51.9131 | 51.8481 | .0650 | .1252 | 158 - 009E |
| 55.0000 | 54.9799 | .0201 | .0366 | 149 - 0095 |
| 58.2705 | 58.0993 | .1712 | .2938 | 141 - 008D |
| 61.7354 | 61.5940 | .1414 | .2291 | 133 - 0085 |
| 65.4064 | 65.5360 | -.1296 | .1982 | 125 - 007D |
| 69.2957 | 69.4237 | -.1281 | .1848 | 118 - 0076 |
| 73.4162 | 73.1429 | .2733 | .3723 | 112 - 0070 |
| 77.7817 | 78.0190 | -.2373 | .3051 | 105 - 0069 |
| 82.4069 | 82.7475 | -.3406 | .4133 | 99 - 0063 |
| 87.3071 | 87.1489 | .1581 | .1811 | 94 - 005E |
| 92.4986 | 92.0449 | .4537 | .4905 | 89 - 0059 |
| 97.9989 | 97.5238 | .4750 | .4848 | 84 - 0054 |
| 103.8262 | 103.6962 | .1300 | .1252 | 79 - 004F |
| 110.0000 | 110.7027 | -.7027 | .6388 | 74 - 004A |
| 116.5409 | 117.0286 | -.4876 | .4184 | 70 - 0046 |
| 123.4708 | 124.1212 | -.6504 | .5268 | 66 - 0042 |
| 130.8128 | 130.0317 | .7810 | .5971 | 63 - 003F |
| 138.5913 | 138.8475 | -.2561 | .1848 | 59 - 003B |
| 146.8324 | 146.2857 | .5467 | .3723 | 56 - 0038 |
| 155.5635 | 154.5660 | .9975 | .6412 | 53 - 0035 |
| 164.8138 | 163.8400 | .9738 | .5908 | 50 - 0032 |
| 174.6141 | 174.2979 | .3162 | .1811 | 47 - 002F |
| 184.9972 | 186.1818 | -1.1846 | .6403 | 44 - 002C |
| 195.9977 | 195.0476 | .9501 | .4848 | 42 - 002A |
| 207.6523 | 210.0513 | -2.3989 | 1.1553 | 39 - 0027 |
| 220.0000 | 221.4054 | -1.4054 | .6388 | 37 - 0025 |
| 233.0819 | 234.0571 | -.9753 | .4184 | 35 - 0023 |
| 246.9417 | 248.2424 | -1.3008 | .5268 | 33 - 0021 |
| 261.6256 | 264.2581 | -2.6325 | 1.0062 | 31 - 001F |
| 277.1826 | 273.0667 | 4.1160 | 1.4849 | 30 - 001E |
| 293.6648 | 292.5714 | 1.0933 | .3723 | 28 - 001C |
| 311.1270 | 315.0769 | -3.9499 | 1.2696 | 26 - 001A |
| 329.6276 | 327.6800 | 1.9476 | .5908 | 25 - 0019 |
| 349.2282 | 356.1739 | -6.9457 | 1.9889 | 23 - 0017 |
| 369.9944 | 372.3636 | -2.3692 | .6403 | 22 - 0016 |
| 391.9954 | 390.0952 | 1.9002 | .4848 | 21 - 0015 |
| 415.3047 | 409.6000 | 5.7047 | 1.3736 | 20 - 0014 |
| 440.0000 | 431.1579 | 8.8421 | 2.0096 | 19 - 0013 |

> 8 Bit
C1
Oktave mit bester Annäherung
H1
Abweichung nimmt zu

gewünschte Frequenz / nächstmögliche Annäherung
F / 256 · n
Differenz: tatsächlich / prozentual
n: Dezimal- Hex

*) Die hier angegebene Festfrequenz F ($2^{21}$ Hz) wird nur von ausgesuchten CD 4526 richtig verarbeitet!

So kann mit fünf Oszillator-Platinen bereits der Eindruck erweckt werden, als wäre für jeden Ton ein eigener Tongenerator vorgesehen (Hammond-Orgel), insbesondere wenn die Teilfaktoren für die Frequenzen je nach Oktave geringfügig verändert werden, so daß bei Akkorden starke Schwebungen entstehen, was bei den phasenstarr arbeitenden Hauptoszillatoren der Orgeln nicht möglich ist. Beliebige Hüllkurven ermöglichen wesentlich mehr Variationen als bei Orgeln, die für jede Taste einen Hüllkurvengenerator benötigen.
Begleitmelodien (bis hin zum Spielen ganzer Musikstücke) sind kein Problem, da ja die Werte ohnehin aus Speichern abgerufen werden, Echo-Effekte sind über Software-Schieberegister ebenso problemlos.
Klangeffekte sind am einfachsten durch Ändern einzelner Schwingungspunkte möglich.
Einfache Hardware-Erweiterungen wie z.B. ein Ringmodulator vor dem Endverstärker, ein Rauschgenerator, A/D-Wandler für Analogeingaben bieten weiter Möglichkeiten.
Die Grenzen des Systems liegen hauptsächlich bei der Rechengeschwindigkeit des Prozessors, meine drei Generatoren lassen sich aber ohne Einschränkung „gleichzeitig" bedienen.
Aber auch der Aufbau nur einer Orgelplatine ermöglicht viele interessante Tonsynthesen.

Klaus Hengstmann

# Farbgrafik-Interface

Mit dem von der Firma Texas-Instruments herausgebrachten LSI-Chip TMS9918 und ein wenig externer Hardware läßt sich ein sehr elegantes Farbvideo-Interface aufbauen. Der Materialaufwand liegt bei etwa 250 DM.
Leider arbeitet der TMS9918 nach dem NTSC-Verfahren, was eine Modifikation des Farbfernsehers zur Folge hat. Diese wird bereits auf dem Markt von einem Hersteller angeboten.

## Die Eigenschaften:

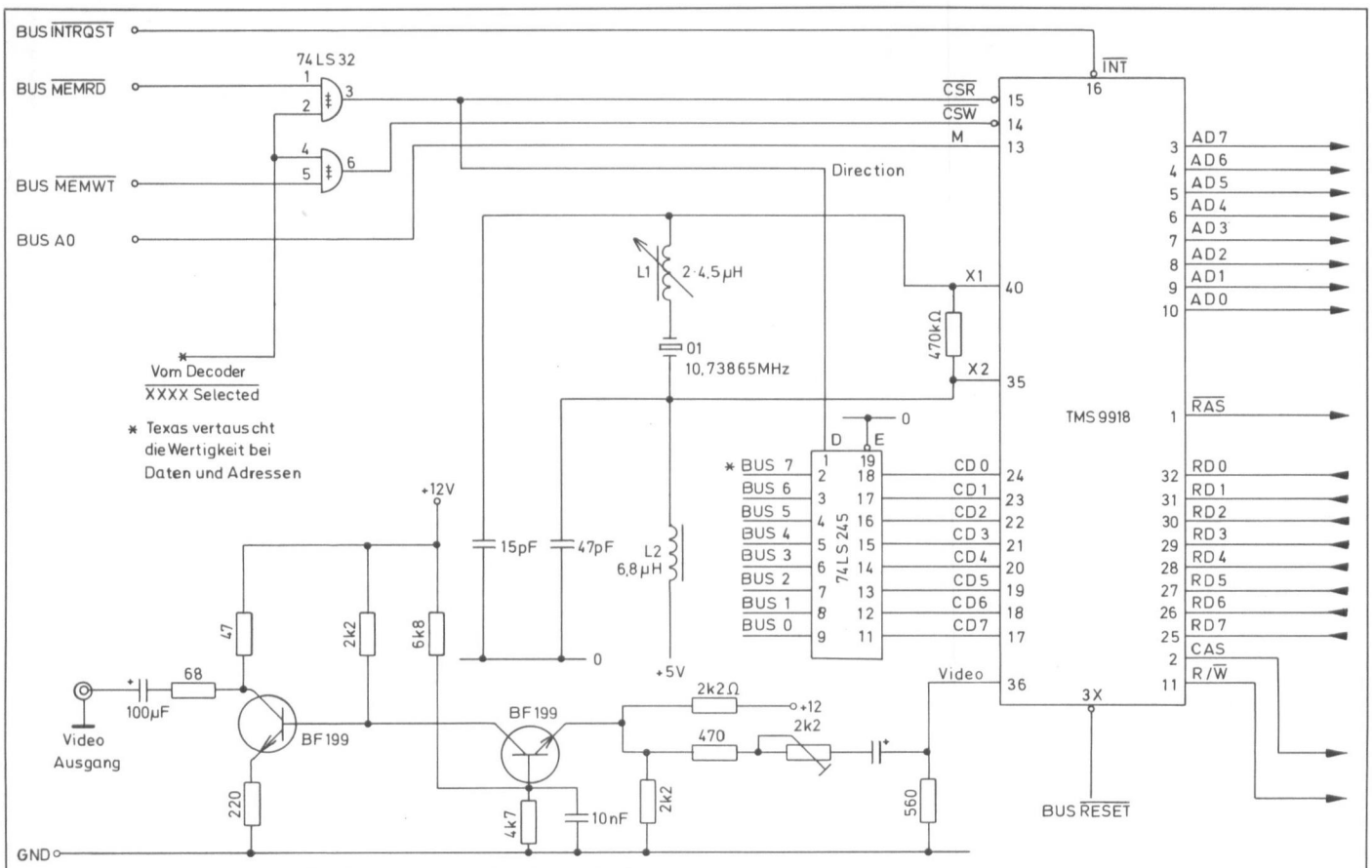
- 16 Farben/8 Graustufen bei S/W
- 256 × 192-Punkte-Auflösung
- 32 freibewegliche Objekte, die gleichzeitig erscheinen können
- 16K-Dyn.-RAM direkt anschließbar
- Verschiedene Betriebsmodi, die auch den Einsatz als Terminal erlauben.
- Minimaler externer Hardwareaufwand

## Die Schaltung:

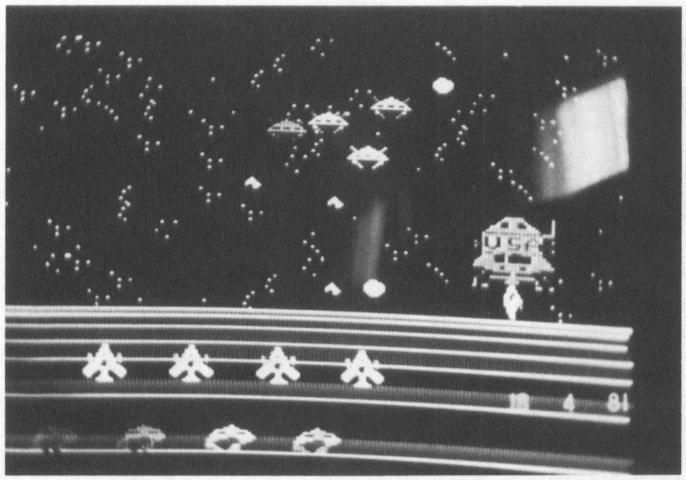
Die Steuerung der dynamischen Schreib-Lese-Speicher und den Refresh übernimmt der TMS9918 selbst. Dadurch entfällt eine Menge Denksport. Eigentlich enthält die Schaltung nur noch zwei kritische Punkte: Der Quarz im Oszillator muß eng toleriert sein, da durch interne Teilung die Farb-Brust-Frequenz davon abgeleitet wird.
Im Videoverstärker sollten rauscharme Transistoren verwendet werden, da im Fernseher noch genug gerauscht wird. Die Ausgangsimpedanz des Verstärkers liegt bei 75 Ω und 1 $V_{SS}$. Bei Verwendung von Koaxialkabel kann die Verbindung zum Fernseher schon mal durch das ganze Wohnzimmer reichen. Zum Ausprobieren können auch BC109C eingebaut werden.
Beim Verfasser fand das gesamte Interface auf einer Europakarte Platz. Der verwendete „Rechner" ist ein handgestrickter Z80 und arbeitet mit dem Farbkontroller über „MEMORY MAPPED I/O". Auf die Einzelheiten der Adressenkodierung wird nicht näher eingegangen.
Außer den Signalen R/$\overline{W}$ und dem Datenbus hat der TMS9918 noch einen Mode-Eingang (M), welcher die Richtung und Art des Datentransfers bestimmt. Hierzu wird die Adresse 0 herangezogen, so daß man zwei Adressen zur Programmierung benötigt.

### Wahrheitstabelle für den Dialog zum TMS9918

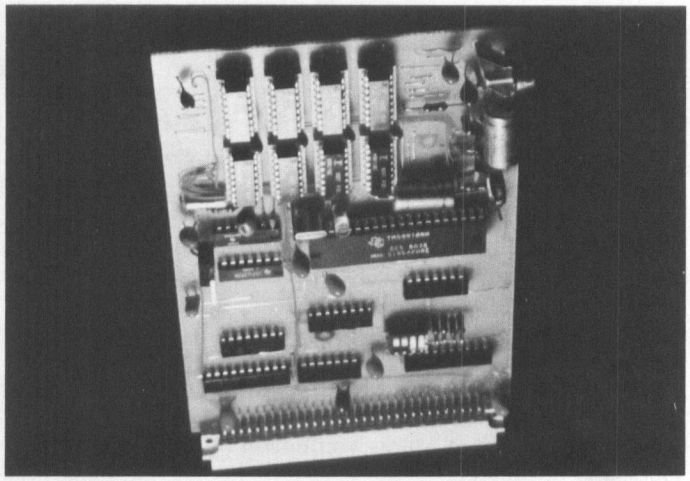
| Operation | W | R | M | Adresse |
|---|---|---|---|---|
| Write Data z. Video-RAM | 0 | 1 | 1 | X X X 0 |
| Write Adrs. z. Video-RAM oder Register | 0 | 1 | 1 | X X X 1 |
| Read Data v. Video-RAM | 1 | 0 | 0 | X X X 0 |
| Read Status Register | 1 | 0 | 1 | X X X 1 |



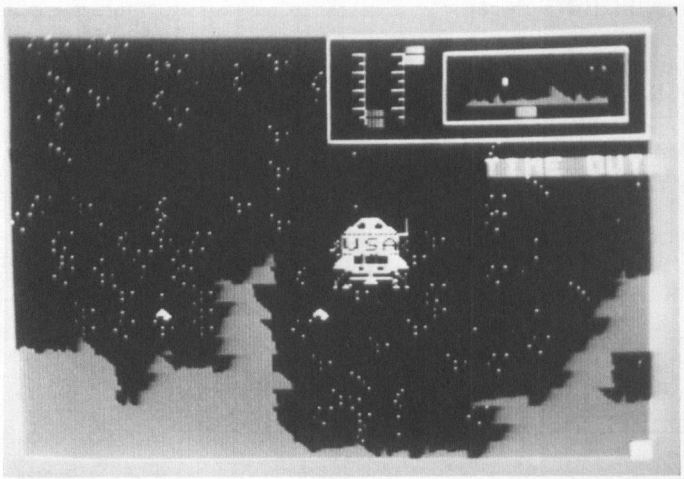

**Schaltbild des Farbgrafik-Interface mit dem TMS9918**



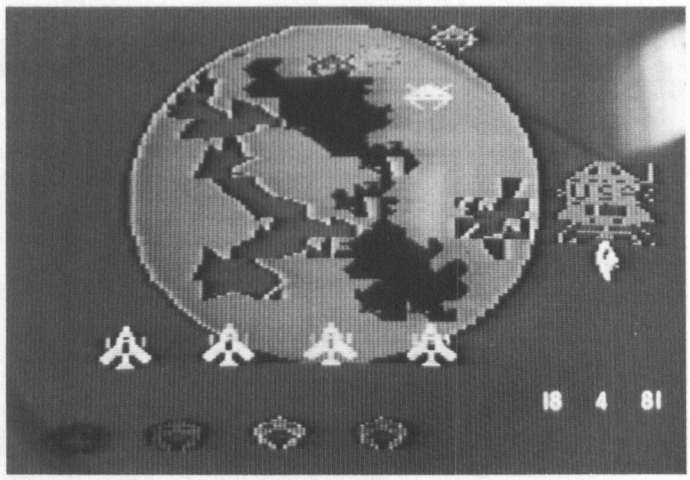
**Spielmöglichkeiten, beispielsweise Raumschiffe, sind in Farbe möglich**



**Gesamte Elektronik auf dem Europaboard für das Interface**



**Ladung einer Raumfähre mit eingeblendetem Kontrollerboard der Fähre**



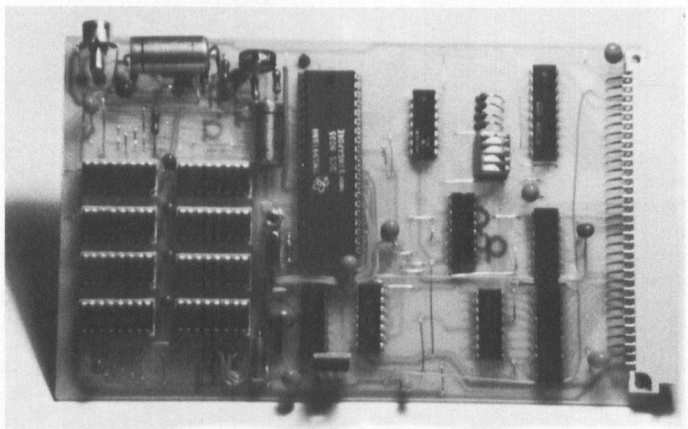
**Möglichkeiten der Darstellung mit transparenten Bildern**

## Die Programmierung:

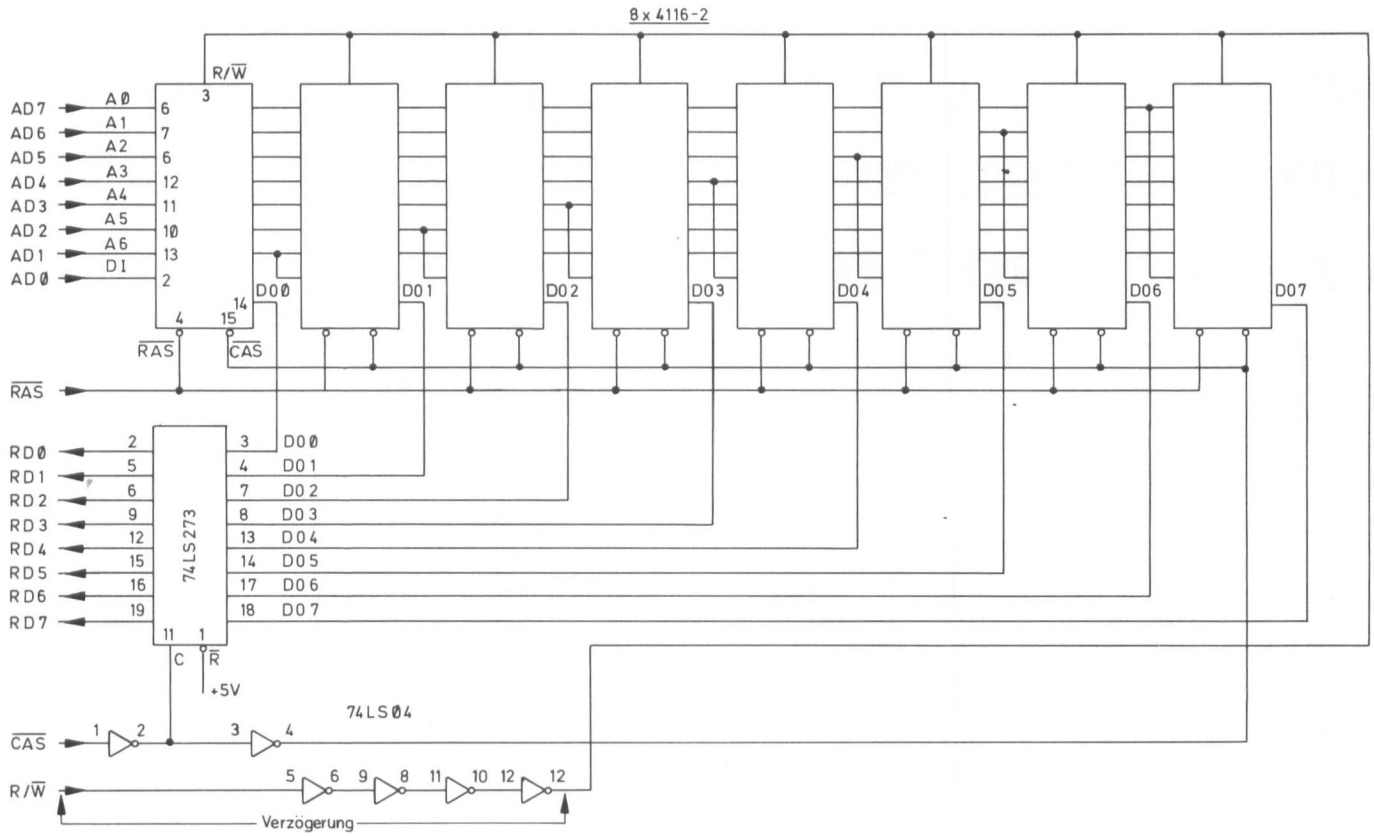
Eine vollständige Beschreibung der Möglichkeiten würde ca. 30 Seiten Text erfordern. Es sei daher auf das Handbuch der Firma Texas-Instruments zum TMS9918 verwiesen. An einigen Beispielen soll nur die Arbeitsweise angesprochen werden.

## Der Bildaufbau:

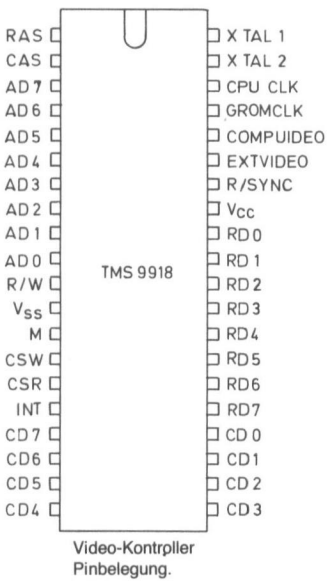
Die Betriebsart mit der höchsten Hintergrundauflösung wird mit „Pattern Mode" bezeichnet. Ein Pattern hat dabei 8 × 8-Bildpunkte. Der gesamte Schirm besteht dann aus 32 × 24-Pattern. Das ergibt 768 Pattern für ein Bild. Jedes Pattern kann in einer „Generatortabelle" definiert werden. Man braucht also acht Byte für ein Pattern. Jedes Bit in einem Byte stellt einen Bildpunkt dar.
Eine 1 läßt den Punkt in der Farbe A, eine 0 in der Farbe B erscheinen. Jeweils acht Pattern werden zu einer Farbgruppe zusammengefaßt. In einer Farbgruppe sind dann alle 1-Signale in der gleichen Farbe und die Nullen in einer anderen zu sehen. Die Farbgruppen werden in der Farbtabelle definiert. Man muß nun für ein Bild nicht alle 256 Möglichkeiten für Patternformen ablegen, es können z.B. nur zwei Pattern den ganzen Schirm füllen, was ein Karomuster sein kann. Lediglich die Namenstabelle ist immer 768 Byte lang, da hier definiert wird, welches Pattern aus der Generatortabelle wo auf dem Schirm erscheinen soll.



**Platine für das Farbgrafik-Interface, vom Autor auf einer Europakarte untergebracht, mit dem TMS9918 und acht dynamischen RAM-Bausteinen**




**Schaltung des dynamischen RAM mit acht Bausteinen und der Refresh-Logik**




Video-Kontrpller Pinbelegung.

Stromversorgungen

| IC TYPE | 0 | +5 | -5 | +12 |
|---|---|---|---|---|
| TMS 9918 | 12 | 33 | | |
| MK 4116 | 16 | 9 | 1 | 8 |
| 74 LS 273 | 10 | 20 | | |
| 74 LS 245 | 10 | 20 | | |
| 74 LS 32 | 7 | 14 | | |
| 74 LS 04 | 7 | 14 | | |

L 1 = ca. 40 Windungen 0,1 Cul auf Spulenkörper Ø 4 mm mit Eisenkern

L 2 = HF Drossel mit 6,8 µH

Signalnamen:

| | |
|---|---|
| AD O – 7 | = Multiplex Bus Address / Data O – 7 |
| DO O – 7 | = RAM Data Out O – 7 |
| RD O – 7 | = Read Data O – 7 |
| BUS O – 9 | = CPU Bus Data O – 9 |
| CD O – 7 | = Controller Data |
| R/$\overline{W}$ | = RAM Read / Write |
| $\overline{CSR}$ | = Controller Read = CPU MEMREAD |
| $\overline{CSW}$ | = Controller Write = CPU MEMWRITE |
| M | = MODE Eingang |
| $\overline{RAS}$ | = ROW-ADR-STROBE |
| $\overline{CAS}$ | = COLL-ADR-STROBE |
| $\overline{BUS\ RESET}$ | = System Reset |
| $\overline{XXXX\ SEL}$ | = Die Karten = Adresse, das Signal ist Low aktiv am Decoder Ausgang. |

Objekte sind in der Vergrößerung 1 wie Pattern definiert. Sie werden auf Punkt × Punkt-Basis positioniert (256 × 192). Das ergibt sehr weiche Begegnungen. Man kann allerdings nur die Farbe einer Eins im Generator-Byte festlegen, die Nullen werden dann durchsichtig, so daß die Objekte vor dem Hintergrund erscheinen.

Das Laden der einzelnen Tabellen gestaltet sich relativ einfach, da der TMS9918 über ein internes 14-bit-Auto-Increment-Adressenregister verfügt. Das heißt, man setzt einmal eine Startadresse auf und braucht dann nur noch weiter die Daten-Byte zu übertragen. Beim Schreiben ist der TMS9918 schnell genug, daß er wie eine Speicherstelle angesprochen werden kann. Nach einem READ sollte der Programmierer ein NOP einfügen, da es ca. 6 µS dauert, bis das nächste Byte zur Verfügung steht. Da man aber in der Regel die Daten auch bearbeiten wird, ist dies nicht unbedingt erforderlich. Ein externer Videoeingang erlaubt es zudem, mit entsprechender Synchronisation ein Videosignal hinter den Hintergrund zu legen. Dort wo dieser „durchsichtig" in der Farbe ist, erscheint dann das Videosignal von außen.

E. Kysely und P. G. Fritz

# Interface für Apple II/ITT 2020 und Christiani-Metallpapierdrucker

## Einleitung:

Die Hauptanforderung an das Interface ist, daß keine Spezial-IC verwendet werden dürfen. Um den Hardware-Aufwand so gering wie möglich zu halten, wird die ganze Steuerungslogik von der Software übernommen. Aus diesem Grund kommt es nicht in Frage, die Stecker für Erweiterungsplatinen als Signalausgang zu benutzen – das Interface wird an den Stecker für Spiele angeschlossen. Da dieser Stecker aber nur wenige Ausgänge hat, wird die Datenübertragung seriell gemacht.

Man könnte meinen, daß dieser Umstand zusammen mit der Logiksteuerung durch Software einen Geschwindigkeitsverlust bringt. Das trifft aber nicht zu, da die Geschwindigkeit vom Drucker weit unter der von diesem Interface liegt, und deswegen der Drucker allein bestimmend für die Druckgeschwindigkeit ist. Das Interface wurde an einem Apple II und an einem ITT 2020 ausprobiert. Es funktionierte in beiden Fällen einwandfrei. Die hier beschriebene Software kann mit dem DOS 3.2.1 und dem DOS 3.3 benützt werden.

## Die Hardware:

Die ganze Interface-Schaltung ist auf einer Platine aufgebaut, und besteht nur aus einem TTL-Shift-Register (74164) und einem CMOS-Buffer-Inverter (4049). Die nötige Spannung nimmt man vom Apple, damit der Drucker nicht losläuft, wenn man den Apple ausschaltet. Die Platine vom Interface wird am besten im Drucker untergebracht.

Als Anschluß braucht man nur zwei 16adrige Flachbandkabel, die an jedem Ende einen Stecker haben, der in eine gewöhnliche IC-Fassung paßt. Außerdem sind am Drucker einige Änderungen durchzuführen. Am Apple sind keine Änderungen nötig.

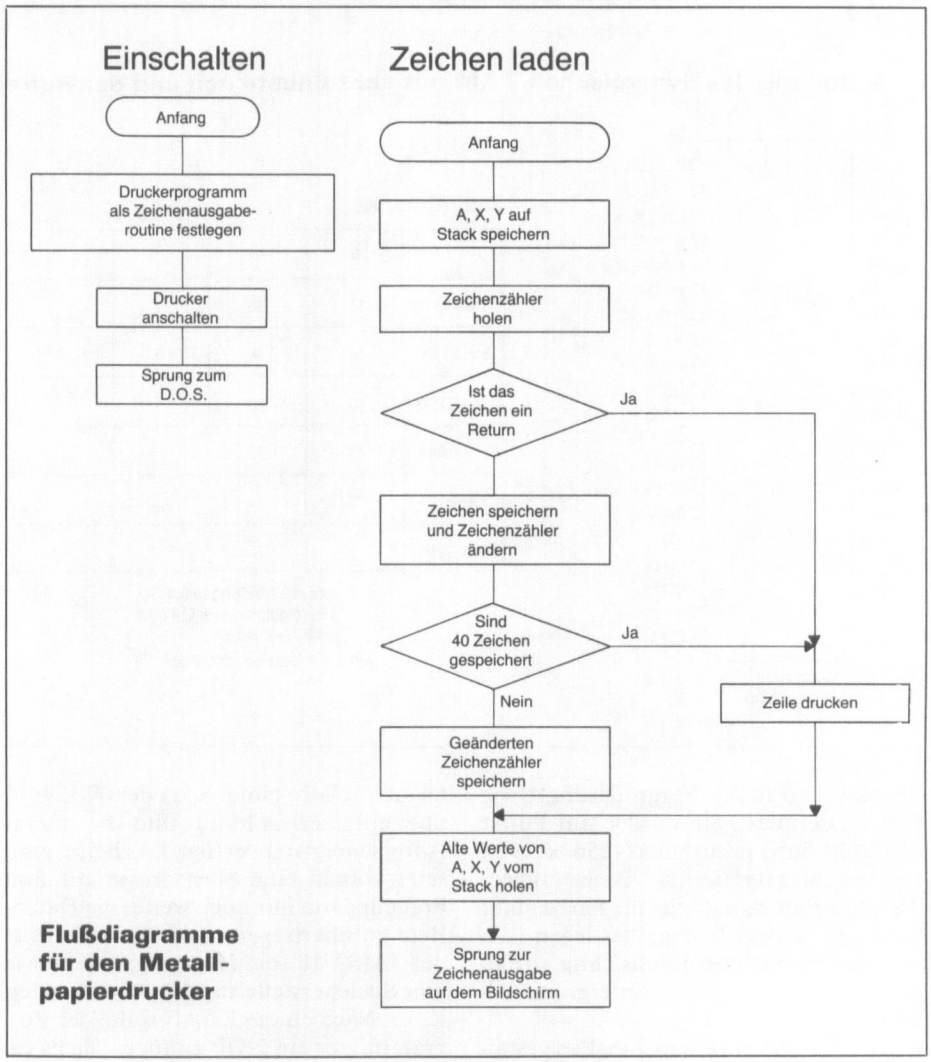
Da der Drucker für 32 Zeichen vorgesehen ist, muß man den Zeitkondensator, der an Bein 6 von IC 555 angeschlossen ist (0,1 µF), gegen einen mit dem Wert von 0,047 µF (47 nF) austauschen. Damit kann der Drucker nach einer erneuten Justierung 40 Zeichen pro Zeile drukken.

Außerdem braucht der Apple ein Signal, das ihm anzeigt, wann der Motor vom Drucker läuft. Deswegen muß eine Brücke gemacht werden, die an den Stekker, Position 1, führt. Diese Änderung ist im beiliegenden Schaltplan des Christiani-Druckers mit M (für Motor) bezeichnet. Diese Änderung muß nicht rückgängig gemacht werden, wenn man den Drucker mit dem Christiani-System benützt, da hier das Bein 1 vom Stecker nicht angeschlossen ist.

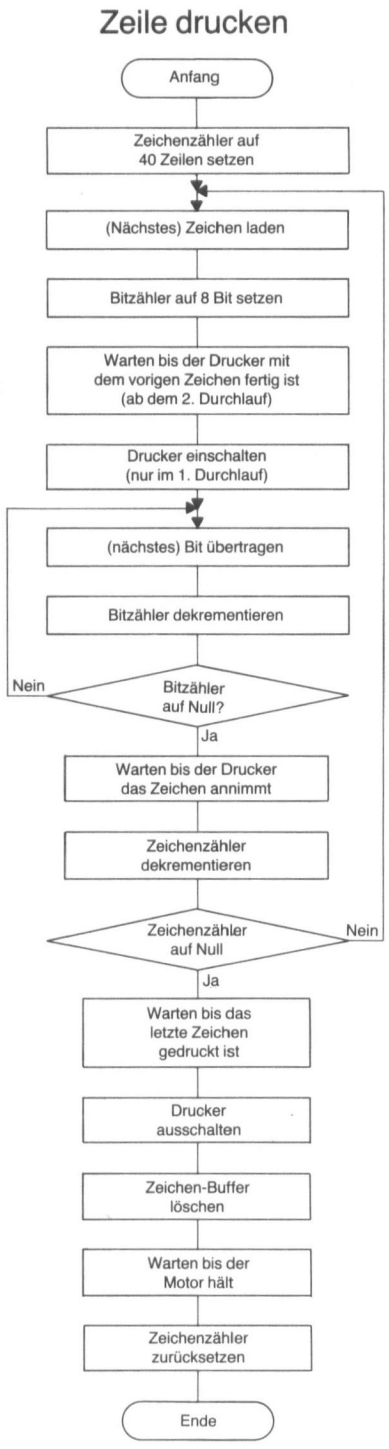
## Die Software:

Die Einzelheiten des Programmes, das für das Interface nötig ist, sind durch das Flußdiagramm und das Maschinensprache-Listing erklärt.

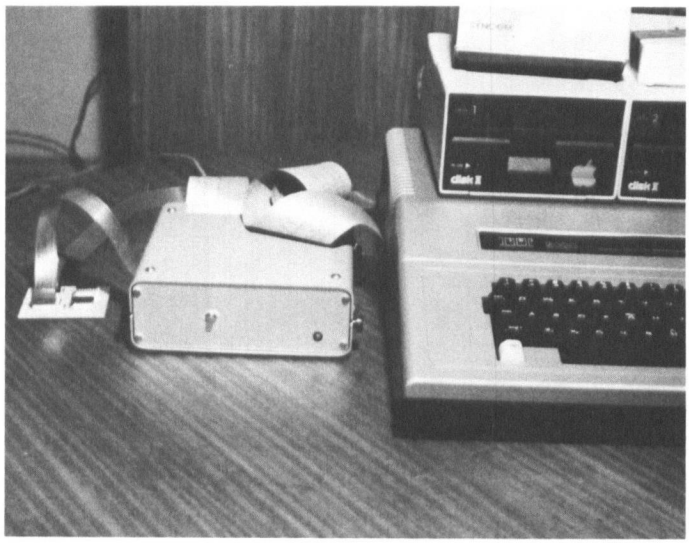
Es sind drei Flußdiagramme beigefügt, die als „Einschalten", „Zeichen laden" und „Zeile drucken" benannt sind. Das mit „Zeile drucken" bezeichnete Flußdiagramm ist ein Teil von „Zeichen laden". Beide wurden nur der Verständlichkeit wegen getrennt. Der Programmteil „Zeile drucken" ist mit demselben



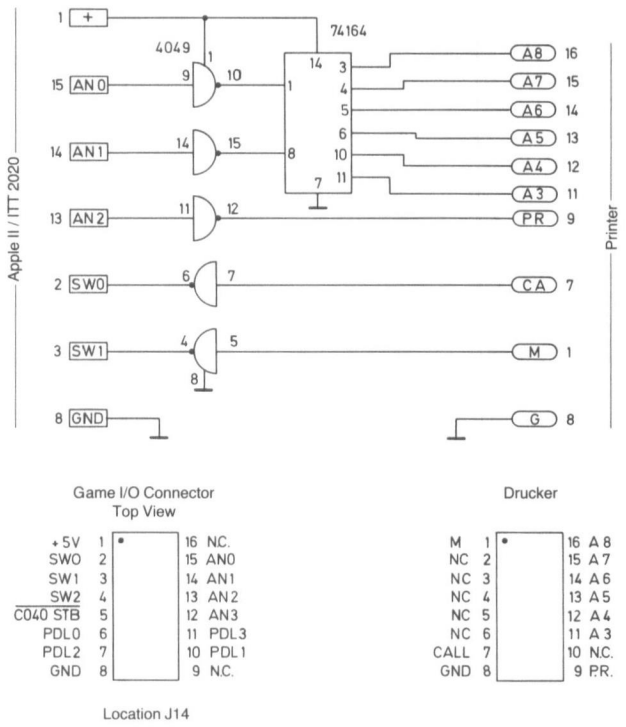

**Flußdiagramme für den Metallpapierdrucker**




**Flußdiagramm für den Metallpapierdrucker**



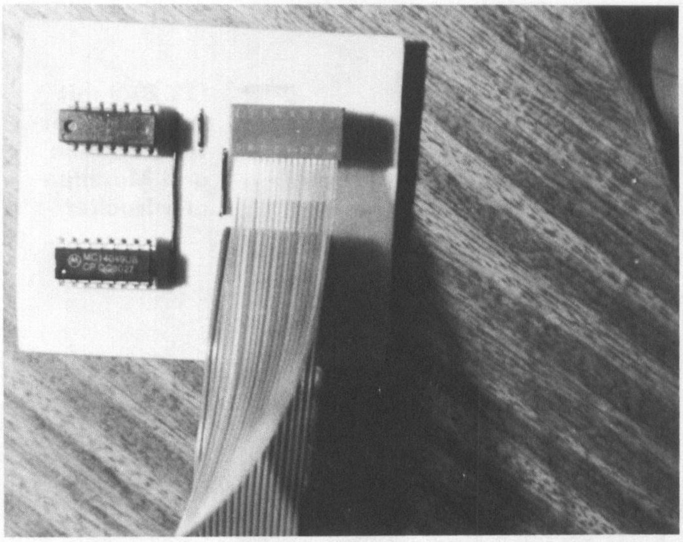
**ITT 2020 mit Apple-Diskettenstationen und Metallpapierdrucker**



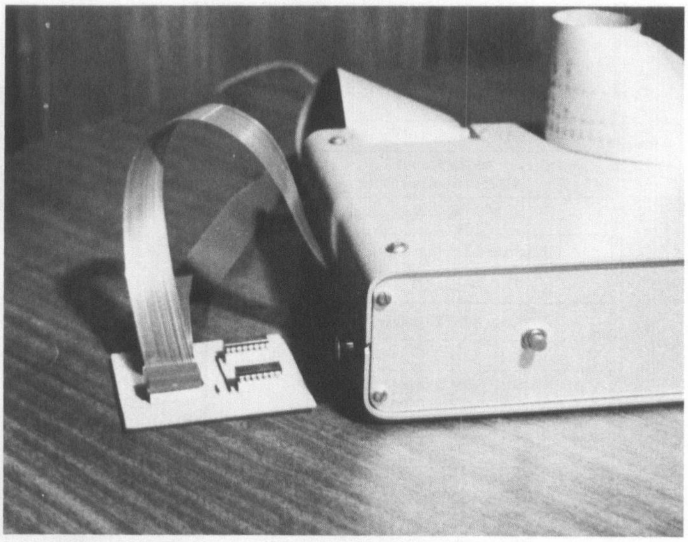

**Schaltung für das Interface zwischen Apple II oder ITT 2020 und dem Metallpapierdrukker**

Namen als ein Block im „Zeichen laden" aufgeführt.
Folglich besteht unsere Software aus zwei getrennten Programmen. Das Programm „Zeichen laden" wird automatisch vom Apple jedesmal aufgerufen, wenn ein Zeichen auf den Bildschirm gedruckt werden soll. Wenn dieses Zeichen kein „RETURN" ist, und die Zeile noch nicht voll ist, dann wird das Zeichen gespeichert und dann an das Ausdruck-Programm vom Apple weitergegeben, damit es am Bildschirm erscheint. Wenn die Zeile schon vorher 39 Zeichen hatte, oder das Zeichen ein RETURN ist, wird vor dem Weitergeben die Zeile ausgedruckt und der Speicher für die neue Zeile leergemacht.

Damit aber dieses Programm immer aufgerufen wird, braucht man das Programm „Einschalten". Dieses Programm sagt dem Apple, wo er das Druckprogramm findet. Der Drucker wird also eingeschaltet, indem man das Programm „Einschalten" aufruft. Dies geschieht wenn man im Monitor „9550G" tippt, oder in BASIC „CALL-27312". Um den



Einfacher Aufbau des Interfaces für den Anschluß



Metallpapierdrucker mit Interface

Drucker wieder auszuschalten, kann man entweder „RESET“ drücken oder in BASIC „PR#O“ tippen.
Es erscheinen alle ausgedruckten Zeichen auf dem Bildschirm. Wenn man dies vermeiden will (die Zeichen nur ausdrucken und nicht auf den Bildschirm geben), kann man eine veränderte Version dieses Programmes schreiben, die diesen Wunsch erfüllt. Dazu muß man an der Stelle 95D3 anstatt des JMP FDFO ein RTS einsetzen.
Dieses Programm wurde für einen Apple II oder ITT 2020 mit 48K-RAM und wenigstens einem Disk-Drive geschrieben. Es ist gleich, ob das System Applesoft oder Integer-BASIC im ROM hat.

## Allgemeines:

Ein Vorteil (oder Nachteil) von diesem Druckprogramm ist, daß der Drucker die CTRL-Zeichen auch ausdruckt, und zwar in derselben Form wie die entsprechenden Buchstaben. Das ist einerseits ein Vorteil, da man versteckte CTRL-Zeichen in einem Listing leicht finden kann. Es kann aber auch ein Nachteil sein, da, wenn der Drucker eingeschaltet ist und man einen Buchstaben auf dem Bildschirm korrigiert, die ausgedruckte Version nicht mit dem Bildschirm übereinstimmt (der linke Pfeil ist ein CTRL-H und der rechte ein CTRL-U). Das stört aber nicht weiter, da man den Drucker meistens dafür benützt, Listings von Programmen zu machen oder Daten programmgesteuert auszudrucken, und in diesen Fällen kommen keine CTRL-Zeichen vor, die stören.
Damit das BASIC das Druckerprogramm nicht zerstört, ist es nötig, nach dem Laden dieses Programmes **HIMEM:-27312** zu tippen.

```
** DRUCKER'81  VON E.KYSELY + P.FRITZ **
*9550LLLLL

9550-   A9 60      LDA   #$60
9552-   85 36      STA   $36
9554-   A9 95      LDA   #$95
9556-   85 37      STA   $37
9558-   AD 50 C0   LDA   $C050
955B-   4C 51 A8   JMP   $A851
955E-   00         BRK
955F-   28         PLP
9560-   8D 5E 95   STA   $955E
9563-   48         PHA
9564-   8A         TXA
9565-   48         PHA
9566-   98         TYA
9567-   48         PHA
9568-   AC 5F 95   LDY   $955F
956B-   AD 5E 95   LDA   $955E
956E-   C9 8D      CMP   #$8D
9570-   F0 0C      BEQ   $957E
9572-   99 D6 95   STA   $95D6,Y
9575-   88         DEY
9576-   F0 06      BEQ   $957E
9578-   8C 5F 95   STY   $955F
957B-   4C CE 95   JMP   $95CE
957E-   A0 08      LDY   #$08
9580-   B9 FF 94   LDA   $94FF,Y
9583-   AE 58 C0   LDX   $C058
9586-   A2 08      LDX   #$08
9588-   8E 5E 95   STX   $955E
958B-   AE 61 C0   LDX   $C061

958E-   30 FB      BMI   $958B
9590-   AE 5C C0   LDX   $C05C
9593-   2A         ROL
9594-   B0 05      BCS   $959B
9596-   AE 59 C0   LDX   $C059
9599-   90 03      BCC   $959E
959B-   AE 58 C0   LDX   $C058
959E-   AE 5A C0   LDX   $C05A
95A1-   AE 5B C0   LDX   $C05B
95A4-   CE 5E 95   DEC   $955E
95A7-   D0 EA      BNE   $9593
95A9-   2A         ROL
95AA-   AE 61 C0   LDX   $C061
95AD-   10 FB      BPL   $95AA
95AF-   C8         INY
95B0-   D0 CE      BNE   $9580
95B2-   AE 61 C0   LDX   $C061
95B5-   30 FB      BMI   $95B2
95B7-   AE 5D C0   LDX   $C05D
95BA-   A0 08      LDY   #$08
95BC-   A9 20      LDA   #$20
95BE-   99 FF 94   STA   $94FF,Y
95C1-   C8         INY
95C2-   D0 FA      BNE   $95BE
95C4-   AD 62 C0   LDA   $C062
95C7-   10 FB      BPL   $95C4
95C9-   A0 28      LDY   #$28
95CB-   8C 5F 95   STY   $955F
95CE-   68         PLA

[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
```

Günter Prokoph

# Hochauflösende Grafik für Z-80-Computersysteme

## Einleitung

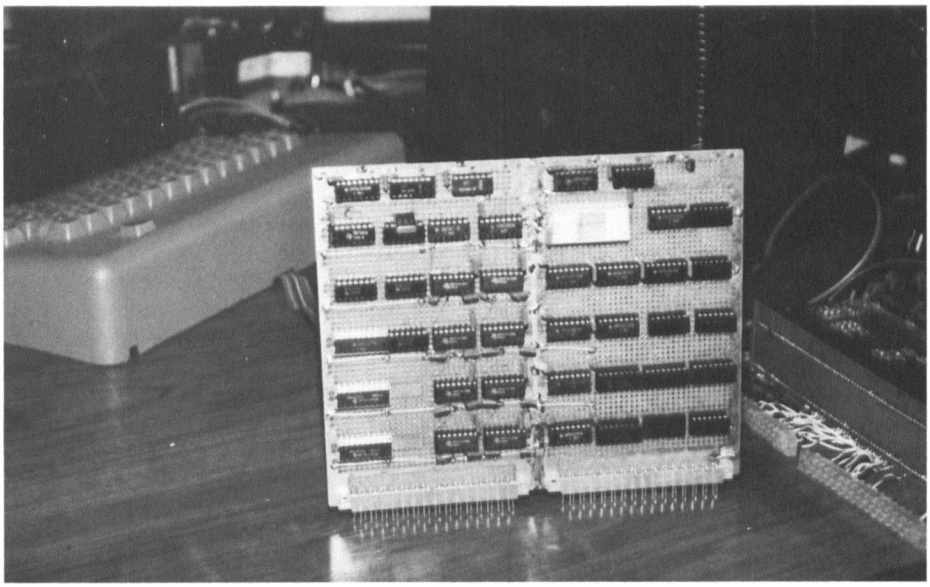
Bei der Darstellung von Computergrafiken mittels des vorhandenen, teilweise auch um Grafikzeichen erweiterten Charaktergenerators, kommt man bald an die Grenze des Machbaren. Dann hegt jeder Computerbesitzer den Wunsch nach einer hochauflösenden Grafik, die es gestattet, auch feiner strukturierte Bilder auf einen Fernsehmonitor zu bringen. Dieser Beitrag beschreibt eine Grafikplatine, mit der es möglich ist, 512 × 256 Punkte einzeln zu setzen oder rückzusetzen. Außerdem können die eingebauten Speicherbausteine (16K × 8) auch als normales RAM verwendet werden.

## Der Aufbau

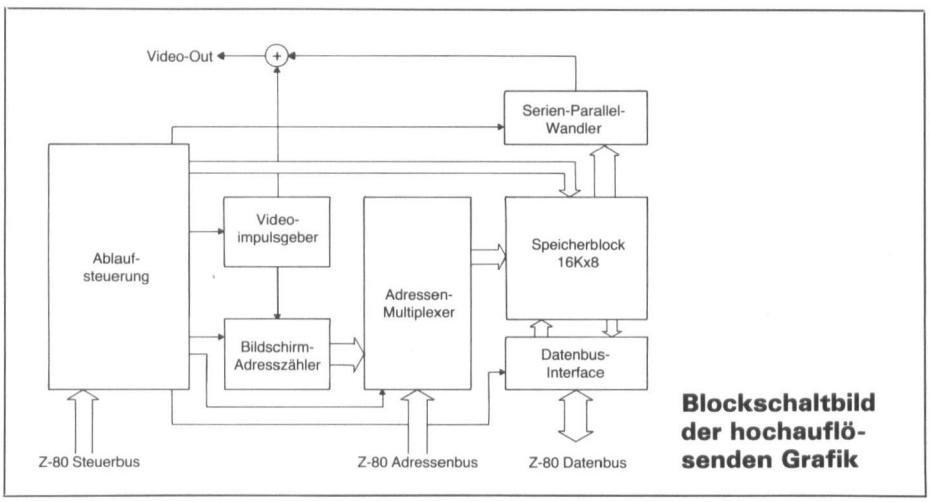
Das Blockschaltbild zeigt den prinzipiellen Aufbau der Schaltung. Der RAM-Block wird mit Steuerimpulsen aus der Ablaufsteuerung versorgt, die sich durch Verknüpfung der Z-80-Steuerbussignale mit einem intern erzeugten 20-MHz-Takt ergeben. Eine Adressenmultiplexschaltung sorgt dafür, daß zur richtigen Zeit entweder die von der CPU erzeugten oder die Bildschirm-Adressenbit an den Speicherbausteinen anliegen. Die Bildschirmadresse wird durch den Bildschirm-Adressenzähler generiert, der durch den Videoimpulsgeber synchronisiert wird. Der Videoimpulsgeber liefert ferner die Synchronimpulse, die zusammen mit dem sich am Ausgang des Parallel-Serien-Wandlers ergebenden Bildsignale dem TV-Monitor zugeführt werden. Eine Interfaceschaltung sorgt für den korrekten Datenaustausch mit dem Z-80-Datenbus.

## Die Ablaufsteuerung

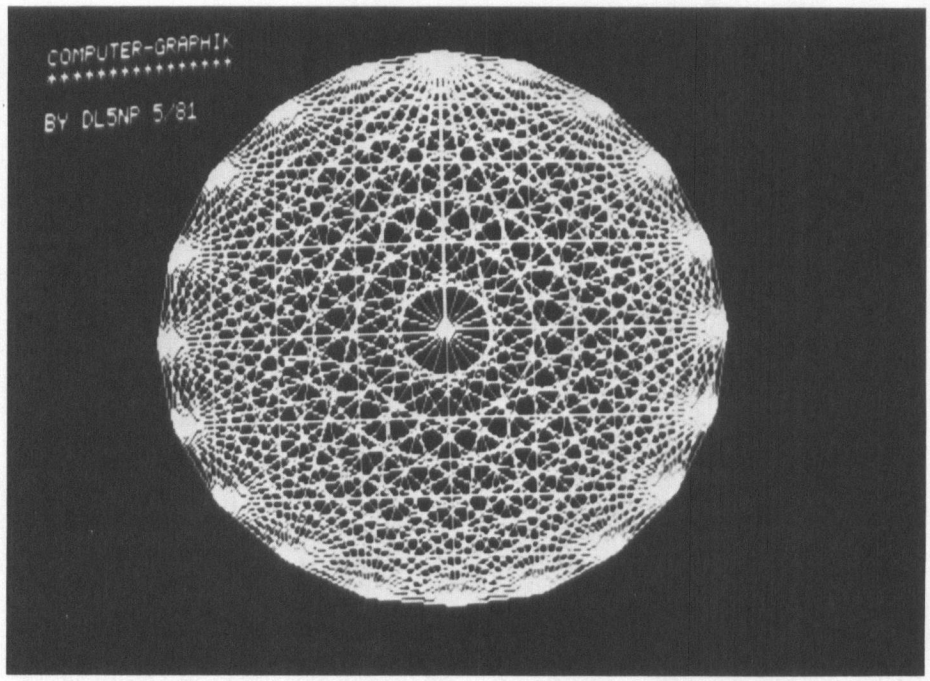
Um mit 512 Bildpunkten eine komplette Fernsehzeile ausfüllen zu können, wurde eine Punktfolgefrequenz von 10 MHz gewählt. Damit ergibt sich eine Zeitdauer von 51,2 µs/Zeile. Betreibt man acht Speicher parallel, so ergibt sich eine Speicherauslesefrequenz von 1,25 MHz (800 ns). Ein Studium des Datenblattes des verwendeten RAM-Bausteins 4116-2 zeigt, daß sich in 800 ns zwei komplette Speicherzyklen unterbringen lassen. Daher liegt der Gedanke nahe, die ersten 400 ns als Lesezyklus zum Auslesen des Bildsignals zu benutzen, während der zweite Zeitabschnitt für einen eventuellen Schreib- oder Lesezugriff des Mikroprozessors zur Verfügung steht. Auf diese Weise ist auch das Refreshproblem der dynamischen RAM gelöst, da die Bildschirmadresse genügend oft alle zum Auffrischen benötigten Werte annimmt. Das Taktdiagramm zeigt den zeitlichen Bezug der einzelnen Signale. Benötigt wird ein Zeitraster im 50-ns-Abstand. Damit ergibt sich die Taktfrequenz des Mutteroszillators (IC1c, IC1d) zu 20 MHz. Sie wird einem 16er-Zähler 74LS163 zugeführt. An dessen Ausgang Q läßt sich nach Invertierung durch IC 2d der 10-MHz-Auslesetakt für das Parallel-Serienwandler-Schieberegister 74LS165 (IC20) gewinnen. Am Carry-Out-Pin des Zähler steht ein 1,25-MHz-Signal an, das den Anfang eines neuen Schirmauslesevorgangs definiert. Es wird über Inverter IC2e einer Kette von zwei Schieberegistern zugeleitet. An den Ausgängen $Q_A$ bis $Q_H$ dieser Bausteine entstehen 50 ns breite und jeweils um 50 ns versetzte Impulse, die das Zeitraster der Ablaufsteuerung bestimmen. Das zweite Schieberegister (IC7) wird nur aktiviert, wenn der Prozessor gerade einen



**Aufgebaute Platine für die hochauflösende Grafik**



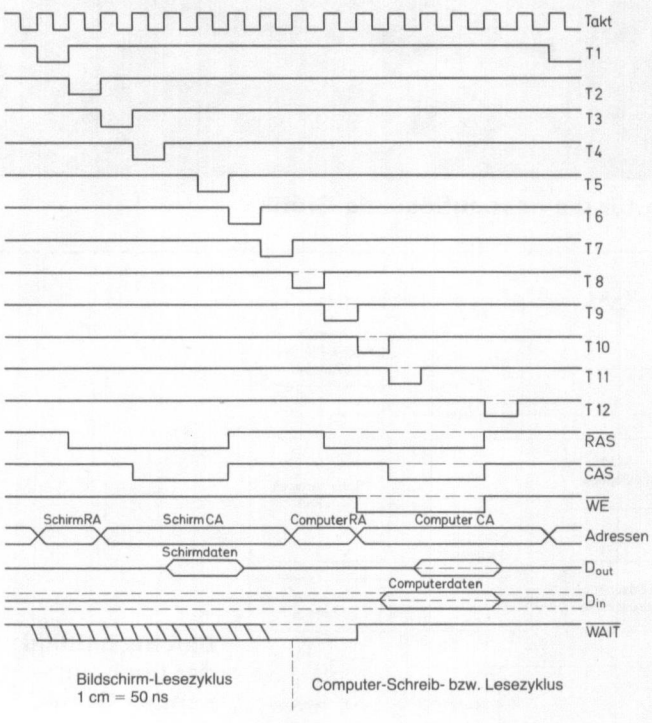

**Blockschaltbild der hochauflösenden Grafik**



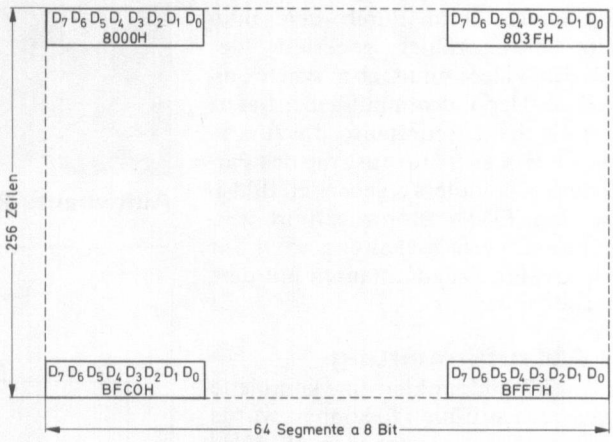

Schreib- oder Lesezyklus beginnen will. Dies ist dann der Fall, wenn neben der richtigen Adresse am Adressenbus (dekodiert durch IC14), auch das Z-80-Signal MEMRQ ein 0-Signal annimmt. Es wird dann das D-Flipflop IC18a gesetzt, welches dafür sorgt, daß der Prozessor ein WAIT-Signal erhält und somit Wartezyklen einfügt, bis er gültige Daten annehmen oder abgeben darf. Außerdem wird, wenn auch etweder $\overline{RD}$ oder $\overline{WR}$ vorhanden sind, über die Gatter 8a, 8d, 2c und 13b ein Computerzyklus eingeleitet (Schieberegister IC7 wird angesteuert). Das $\overline{WAIT}$-Flipflop IC18a wird durch das Auftreten des Impulses T10 zurückgesetzt, so daß der Z-80 seinen Speicherzyklus beenden kann. Die Steuerimpulse für den RAM-Block und die Adressenmultiplexer werden durch RS-Flipflop erzeugt, die durch die passenden Zeitrastersignale gesetzt bzw. rückgesetzt werden. Der WE-Impuls wird nur generiert, wenn der Prozessor einen Writezyklus anfordert. Dieses geschieht über die Gatter 12d und 11c. Die Ablaufsteuerung liefert außerdem noch die Signale zum Durchsteuern der Ein- bzw. Ausgabepuffer (IC31, IC32) sowie des Datenübergaberegisters IC30, das Shift/Load-Signal für das Parallel-Serien-Wandler-Schieberegister und den Takt für den Bildschirmzähler (IC22, IC23).

## Die Videoaufbereitung

Kernstück der Videoaufbereitung ist der hochintegrierte Videoimpulsgeberbaustein S178 bzw. S178a der Firma Siemens. Er benötigt einen 1MHz-Takt mit Tastverhältnis 1:1, um am Ausgang normgerechte Signale zu produzieren. Dieser Takt wird aus der Mutterfrequenz durch den 10er-Teiler 74LS162 (IC3) und anschließende Frequenzhalbierung (IC4a) erzeugt. Abweichend von der bestehenden Fernsehnorm wird hier bewußt auf den Zeilensprung verzichtet (Programmierung auf 624 Zeilen). Dies ergibt eine vollkommen stehende Bildwiedergabe, die bei 625 Zeilen nicht erreicht werden konnte (vertikales Flimmern mit 25Hz). Verwendet man die Version S178, muß Pin 19 als Eingang (H) an +5 V geschaltet werden. Beim S178a bleibt der genannte Anschluß offen, da es sich um einen Ausgang handelt (V). Alle benützten Ausgänge des MOS-Chip sind über IC16 gepuffert, um schädliche Belastungen zu vermeiden. Die Dioden (1N4148) begrenzen die Ausgangsspannung zu negativen Werten hin, um eine Gefährdung der TTL-Bausteine auszuschließen. Das Synchronsignal (IC6 Pin 6) wird direkt auf die Videoaddierstufe




**Taktdiagramm zur hochauflösenden Grafik**




**Bildschirmposition-adressbezug am Beispiel der Adressenbelegung von 8000H bis BFFFH**

(BC237) geführt. IC18b und IC19 zählen nach Beendigung des vertikalen Synchronimpulses 32 Zeilen ab und bestimmen so den Beginn des Bildinhaltes. Über IC4b, IC17d und IC17c wird dann der Weg des Ausgangssignals des Parallel-Serien-Wandlers freigegeben. IC17c sperrt es jedoch auch in den horizontalen Austastlücken (Verknüpfung mit dem A-Signal). Das D-Flipflop (IC21a) erzeugt den Rücksetzimpuls für den Zeilenadressenzähler, dessen Taktfrequenz sich aus dem H-Impuls ableitet.
Hat der Zeilenzähler den Wert 255 erreicht, wird mit dem nächsten Takt über IC21b, IC4b und IC17d das Bildsignal erneut gesperrt (unteres Ende des aktiven Bildinhaltes). Der Bildsegmentzähler (IC22, IC23) erhält seinen Takt (1,25 MHz) aus der Ablaufsteuerung und wird durch den H-Impuls zurückgesetzt.

## Das Adressenmultiplexing

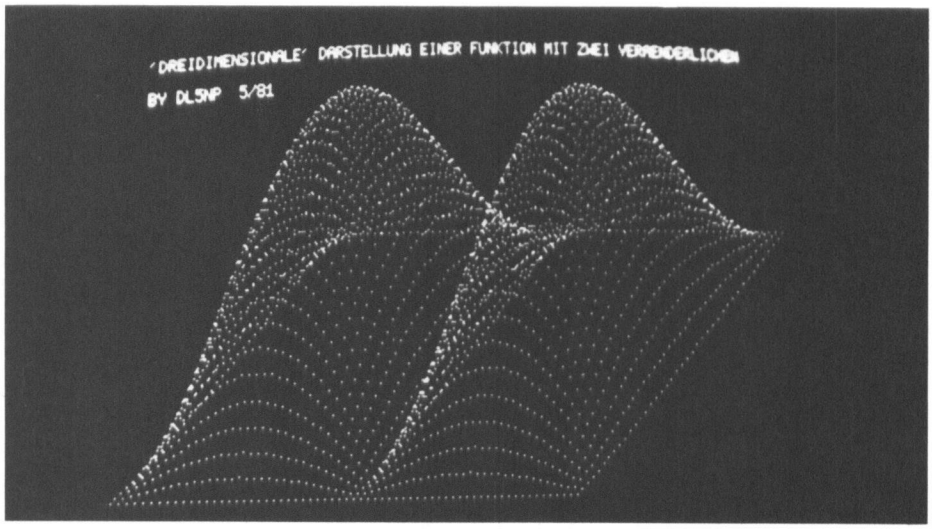
Dynamische RAM vom Typ 4116 benötigen die Adresse (14 bit) in einer zweifach gemultiplexten Form. Die Reihenadresse RA umfaßt die sieben niederwertigen, die Spaltenadresse CA die sieben höherwertigen Bit. Beide Teiladressen müssen nacheinander an die entsprechenden Pins der Speicher angelegt werden. Strobeimpulse (RAS und CAS) sorgen für die korrekte Übernahme in den integrierten Baustein. Im vorliegenden Fall müssen weiterhin zwei 14-bit-Adressen umgeschaltet werden (Bildschirmadresse und Mikroprozessoradresse). Damit ergibt sich ein 4 : 1-Multiplexing. Diese Aufgabe läßt sich mit vier Bausteinen vom Typ 74LS257 (IC26 – IC30) lösen. Zwei 2 : 1-Multiplexer werden am Ausgang parallelgeschaltet. Welcher von beiden aktiv wird, bestimmt die Spannung am Pin „OC" (Output Control). Durch geeignete Impulse aus der Ablaufsteuerung können die vier Teiladressen in der gewünschten Reihenfolge an den Speicheradressbus gelegt werden.

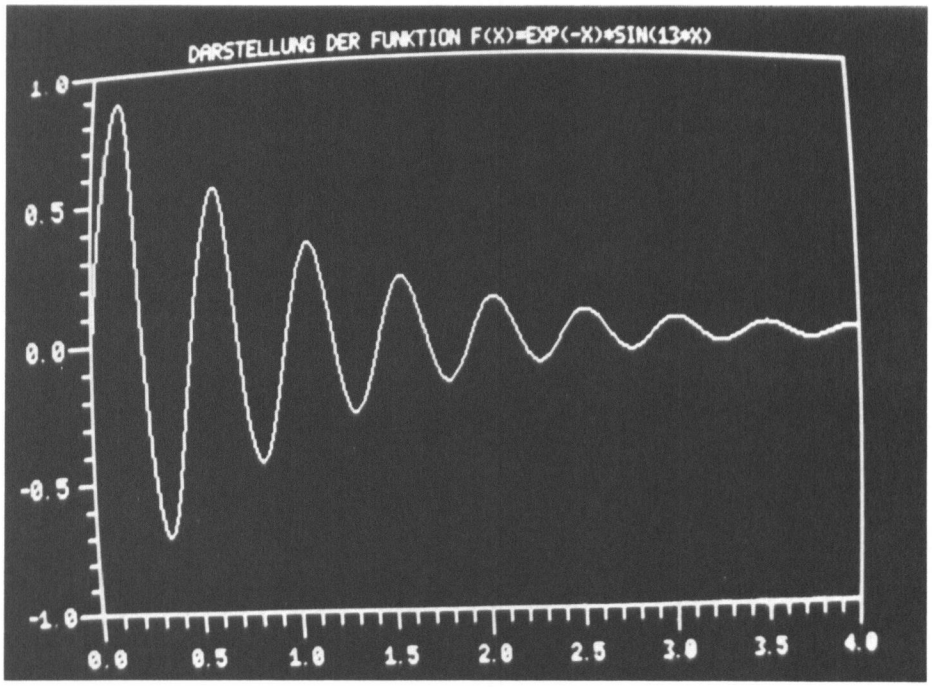
## Das Businterface

Zwei Schaltkreise vom Typ 74LS240 bilden die Verbindung zum Z-80-Datenbus. Die Signale an „1G" bzw. „2G" bestimmen die Übertragungsrichtung. Die Eingangsdaten werden direkt den $D_{in}$-Pins der acht Speicherbausteine zugeführt. Da die Ausgangsdaten nur begrenzte Zeit an den $D_{out}$-Pins der RAM anliegen, werden sie in einem 8fach-D-Register zwischengespeichert, um einen sicheren Lesezyklus des Z-80 sicherzustellen.

## Die zusätzliche Adressenbit

Am IC12b Pin 5 kann ein zusätzlicher Adressendekoder angeschlossen werden (z.B. für paged Memory). Er muß bei gültiger Adresse ein O-Signal an Punkt X liefern.



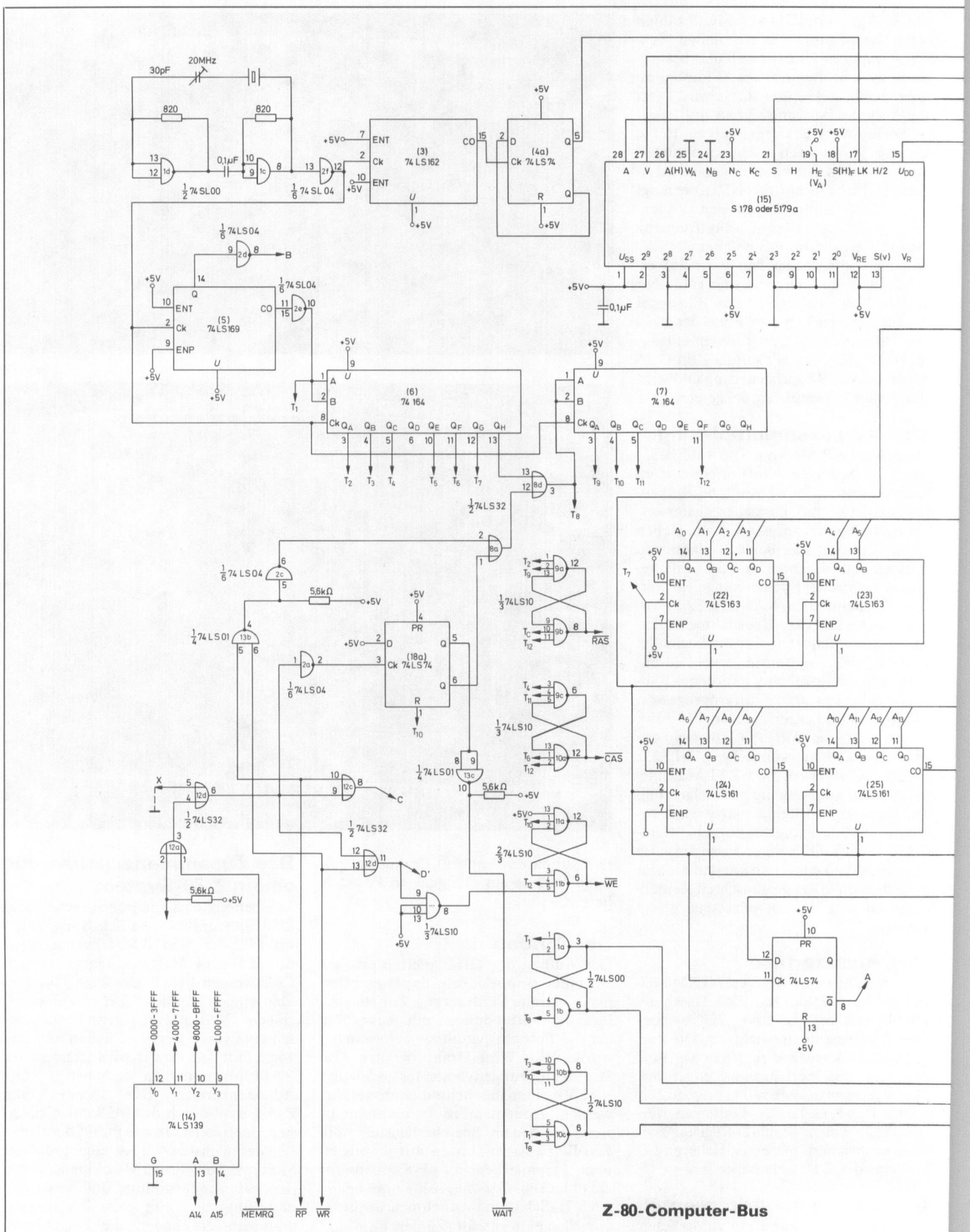



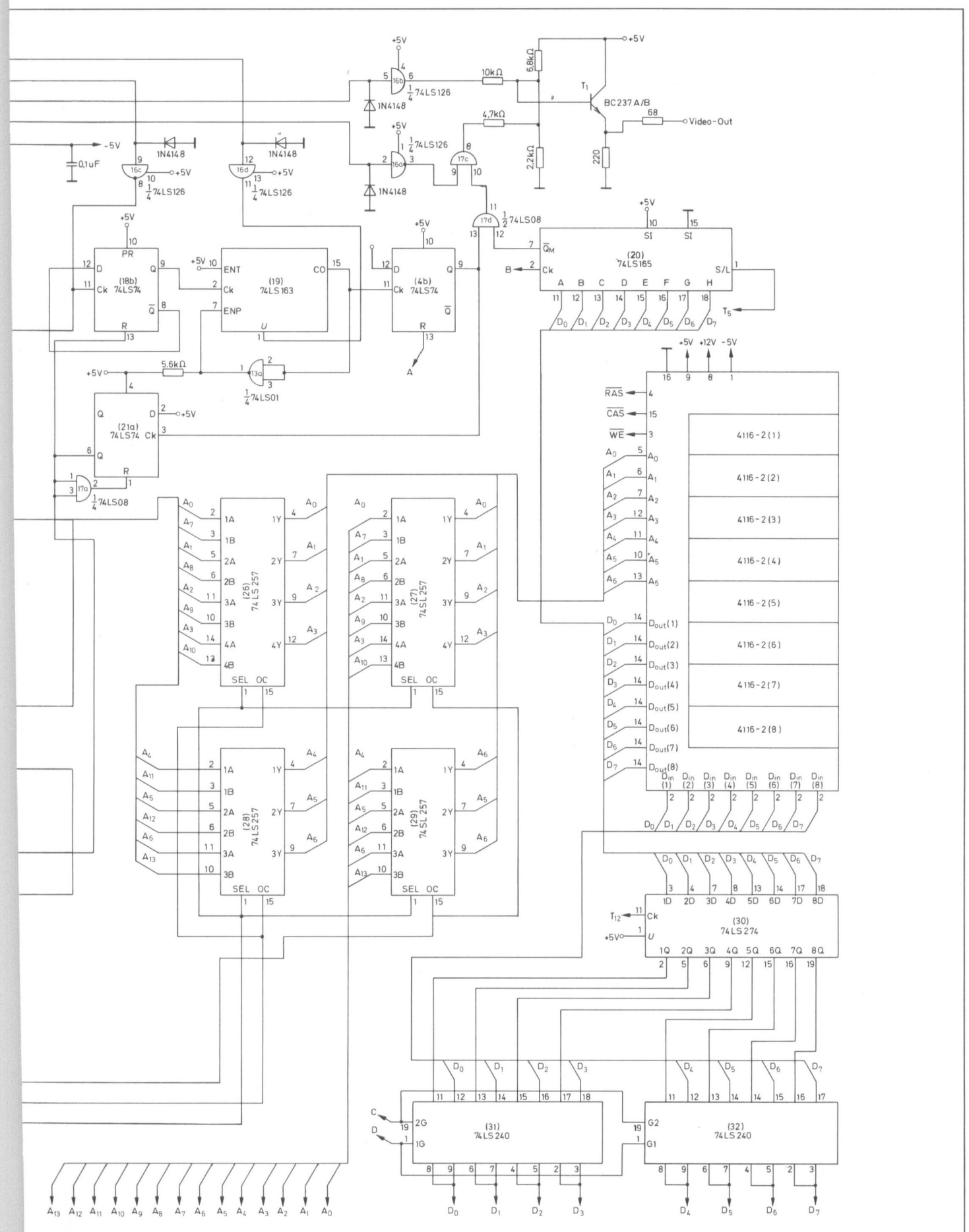

## Der Aufbau

Der Aufbau der Grafikplatine erfolgte auf einer Doppeleuropa-Experimentierplatte in freier Verdrahtung. Zur Herstellung der Verbindungen mit Ausnahme der Betriebsspannungsversorgungen wurde Wire-Wrap-Draht benutzt. Die Stromzuführungen (besonders + 5 V und + 12 V) sollten möglichst dick ausgeführt werden. Außerdem ist es empfehlenswert, bei jedem Speicherbaustein alle drei Betriebsspannungen mit jeweils einem keramischen 0,1-µF-Kondensator abzublocken. Gleiches sollte man bei jeder TTL-Schaltung vornehmen, die bistabile Elemente enthält (Zähler, Flipflops, Register).

## Das Zusammenschalten mit einem Z-80-System

Alle benötigten Signale entsprechen dem Z-80-Standardbus. Die Schaltung ist sowohl für den Z-80 (2 MHz) als auch für den Z-80a (4 MHz) geeignet. Je nach CPU werden 1 bis 2 oder 2 bis 3 Waitzyklen eingefügt. Dies führt zu einem geringen Verlust an Verarbeitungsgeschwindigkeit, der aber selten ins Gewicht fällt. Die Grafik wird als normales RAM angesprochen und belegt 16K-Adressenbereich. Der oberste linke Punkt wird durch das MSB der niedrigsten, der unterste rechte Punkt durch das LSB der höchsten Adresse angesprochen. Mit etwas Software in Maschinensprache (Koordinatendarstellung und Vektorgenerierung) und einfachen BASIC-Programmen entstanden die beigefügten Aufnahmen. *(bitte umblättern)*


20MHz
30pF
820
820
0,1µF
1/2 74 SL00
1/6 74 SL 04
(3)
74 LS162
(4a)
Ck 74 LS74
(15)
S 178 oder 5179a
1/6 74LS04
1/6 74SL04
(5)
74LS169
(6)
74 164
(7)
74 164
1/2 74LS32
1/6 74 LS04
1/4 74 LS01
5,6kΩ
(18a)
74LS74
1/3 74LS10
2/3 74LS10
(22)
74LS163
(23)
74LS163
(24)
74LS161
(25)
74LS161
1/2 74LS00
(14)
74 LS139
0000-3FFF
4000-7FFF
8000-BFFF
C000-FFFF
A14
A15
MEMRQ
RP
WR
WAIT
RAS
CAS
WE
Z-80-Computer-Bus


+5V
-5V
0,1 uF
1N4148
1/4 74LS126
10kΩ
6,8kΩ
4,7kΩ
2,2kΩ
220
68
T1
BC237 A/B
Video-Out
1/2 74LS08
(20) 74LS165
(18b) 74LS74
(19) 74LS163
(4b) 74LS74
5,6kΩ
1/4 74LS01
(21a) 74LS74
1/4 74LS08
(26) 74LS257
(27) 74LS257
(28) 74LS257
(29) 74LS257
SEL OC
4116-2(1)
4116-2(2)
4116-2(3)
4116-2(4)
4116-2(5)
4116-2(6)
4116-2(7)
4116-2(8)
+12V
RAS
CAS
WE
(30) 74LS274
(31) 74LS240
(32) 74LS240

Ralf Kuron

# Grafikerweiterung

In CHIP wurde bereits ein Videointerface mit Grafikeigenschaften beschrieben, allerdings nur mit der Einschränkung einer bestimmten Menge festgelegter Zeichen in einem 8 × 8-Punktefeld. Mit Hilfe dieser Erweiterung wird das Videointerface voll grafikfähig. Es bietet nun die Möglichkeit, auf dem Bildschirm in einem 192 × 256 Bildpunkte umfassenden Feld jede beliebige Grafik zu erzeugen, da jeder Punkt einzeln adressierbar ist.

Abgesehen von der Grafik, kann natürlich auch jeder beliebige Schrifttyp realisiert werden.

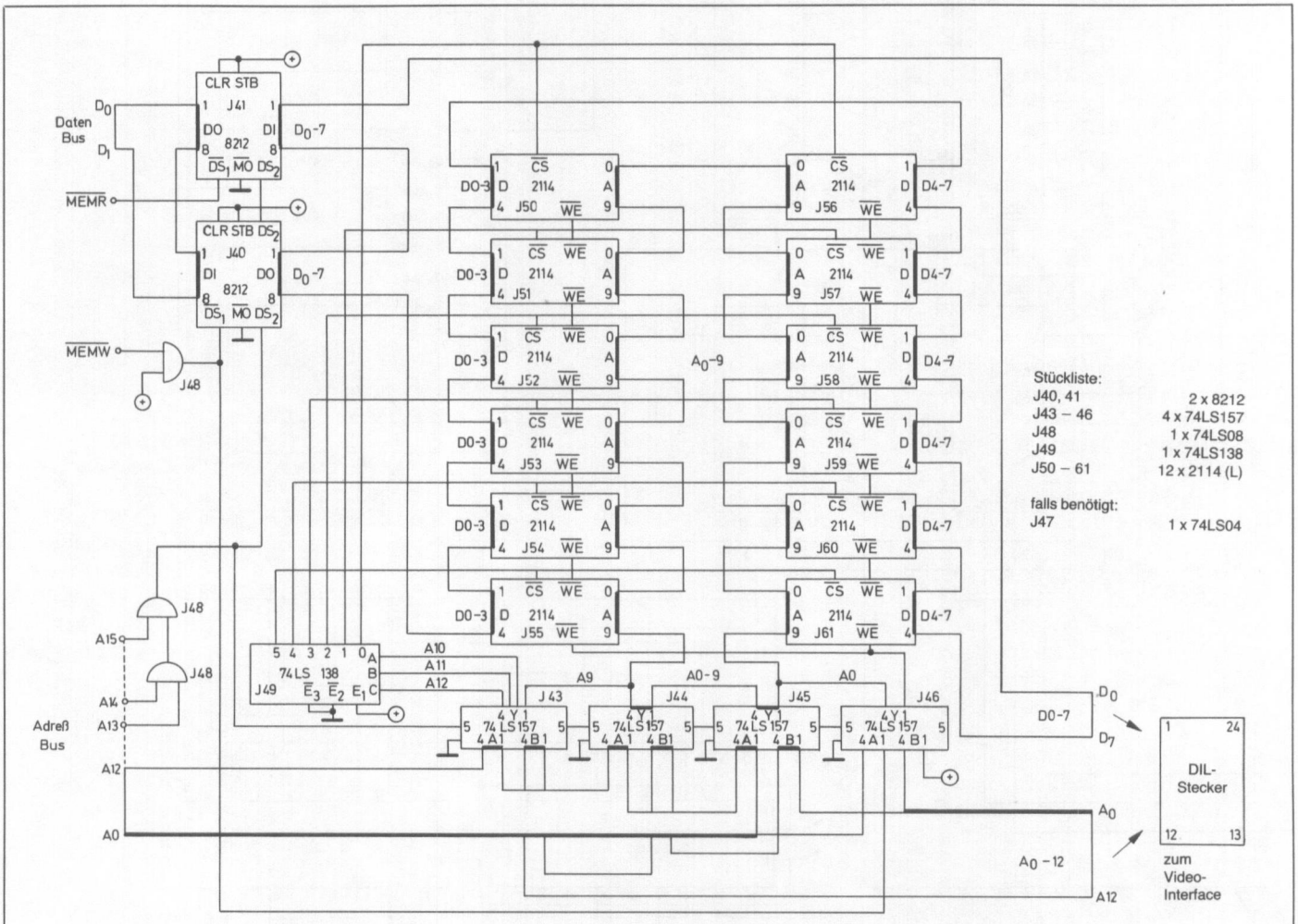
Auf der Videokarte lassen sich die notwendigen Veränderungen relativ einfach vornehmen, indem lediglich einige neue Drahtbrücken gelegt und Verbindungen unterbrochen werden.

## Die Funktionsbeschreibung:

Die eigentliche Erweiterung besteht darin, daß nunmehr jedem der schon bei der ursprünglichen Version möglichen 192 × 256 Bildpunkte (entspricht 24 Zeilen × 32 Zeichen pro Zeile mit 8 × 8-Punktefeldern, worin die Zeichen einbeschrieben werden) genau ein RAM-Datenbit entspricht. Ist also ein Bit gesetzt (≙ 1), so wird der entsprechende Punkt auf dem Bildschirm hell. Bekanntlich wird von der Videointerfacekarte ein 0,75 KByte umfassendes RAM adressiert, das somit die Information liefert, auf welcher Bildschirmposition welches ASCII- (oder Grafik-)Zeichen abgebildet werden soll. Der ASCII-Kode aus dem RAM wird als 7-bit-Teiladresse an das EPROM gelegt, welches die Umsetzung des ASCII-Kodes in das entsprechende Bildmuster leistet. Die Zeichen bei diesem System belegen immer ein Feld von 8 × 8-Bildpunkten. Es sind also mit der dann am Ausgang des EPROM vorhandenen 8 bit langen Zeileninhaltsinformation pro ASCII-(oder Grafik-)Kode-8-Byte (entsprechend acht untereinanderliegenden Zeilen) pro Zeichen im EPROM abgelegt.

Die jeweilige gerade benötigte Zeile des Zeichens liefert der sechsstufige Zähler 74193 als 3-bit-Information im Rahmen des zeilenweisen Aufbaus des Fernsehbildes. Diese 3-bit-Information (bezeichnet als X, Y, Z) vervollständigt somit die Adresseninformation an das EPROM zum ASCII-Kode hinzu.

Möchte man aber auf jeden Bildpunkt freien Zugriff haben, muß ein anderer Weg beschritten werden. Es wird kein 8-bit-Kode, welcher genau einem definierten Zeichen entspricht (ASCII), im RAM abgelegt und von einem EPROM umge-




**Schaltung für die Erweiterung der Grafikmöglichkeiten**

setzt. In ein RAM wird ein Byte eingeschrieben, das dann unmittelbar auf dem Bildschirm dargestellt wird.

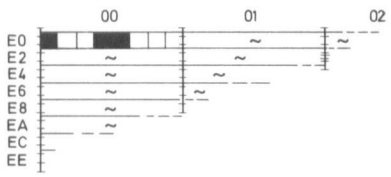
Hierbei entspricht jeder Adresse des RAM genau eine Position auf dem Bildschirm (bestehend aus acht Bildpunkten wegen dem byteorientierten RAM bzw. Mikroprozessor).

Möchte man beispielsweise eine durchgehende Linie am oberen Bildrand erzeugen, so schreibe man in die ersten 32 Speicherstellen (≙ einer Zeile) des RAM je „FFH" ≙ 1111 1111.

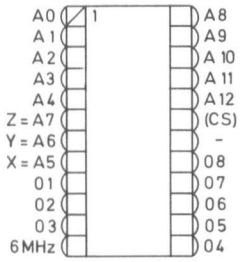
## Der Aufbau:

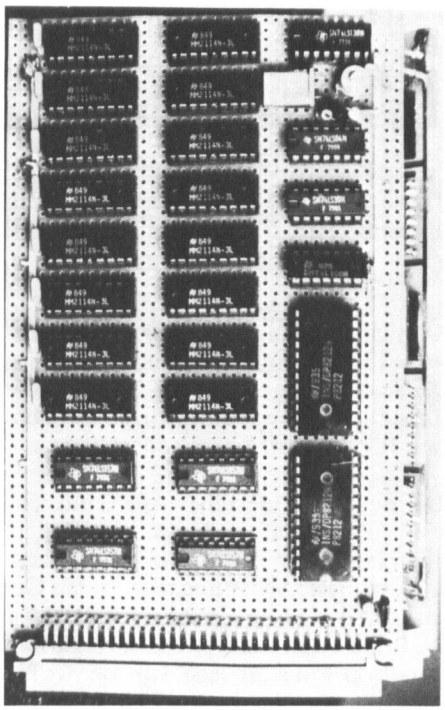
Der Bildwiederholspeicher (RAM) vergrößert sich wegen der höheren Auflösung auf 6 KByte.
Die Adresseninformation für das RAM leitet sich natürlich aus den zehn Leistungen, welche schon für das $^3/_4$ K-RAM auf der Videointerfacekarte vorgesehen waren, sowie den drei Zeilendekodierleitungen X, Y, Z her. Mit den besagten 13 Adressenleitungen ließen sich maximal 8K adressieren, die Zählerstufen zählen jedoch nur bis 6143 D (≙ 6K). 6 KByte entsprechen 49152 einzelnen Bit, und diese Anzahl deckt sich natürlich mit dem 192 × 256-Punkte-Grafikbild.
Im übrigen ist die Funktionsweise gleich geblieben. Der 6K-RAM-Bereich kann sowohl vom µP als auch vom Videointerface her beschrieben bzw. gelesen werden. Der µP hat beim Zugriff auf den Bildwiederholspeicher Priorität.
Die beiden antiparallel geschalteten 8212 (J40.J41) dienen als bidirektionaler Tri-State-Datenbusbuffer. Sie werden nur dann durchgeschaltet, wenn der µP einschreiben bzw. lesen möchte. Solange das Videointerface Zugriff auf den Bildwiederholspeicher hat, sperren J40 und J41. (Für 8212 können auch 74LS244 o.ä. eingesetzt werden!)
Mit Hilfe der Adressenmultiplexer J43, J44, J45, J46 werden entweder der µP-Adressenbus auf das 6K-RAM gelegt oder die Adressenleitungen der Zählerstufen 74193. Die Umschaltung wird durch die Adressenbit A13, A14, A15 des µP bestimmt. In der abgebildeten Version fängt der 6K-RAM-Bereich bei der Adresse E000H an.
Außer den Adressen wird durch die Multiplexer auch die R/$\overline{W}$-Leitung umgeschaltet. Im µP-Zugriff ist die $\overline{NEMW}$-Leitung durchgeschaltet, womit der µP zwischen Lesen und Schreiben umschalten kann. Greift das Videointerface auf das RAM zu, so wird automatisch auf Nurlesebetrieb gearbeitet ($\overline{WE}$ = 1). Der 6K-RAM-Bereich läßt sich gut mit dem Typ 2114 (L) realisieren. Natürlich ist auch jeder andere Speicherbaustein geeignet, sofern er eine Zugriffszeit von maximal 900 ns aufweisen kann!
Auf der Karte kann man außerdem noch den benötigten 18-MHz- bzw. 6-MHz-Quarzoszillator unterbringen (J47), sowie bei Bedarf eine vollständige Dekodierung für das 1K-RAM auf der Videointerfaceplatine.
Denkbar wäre auch ein Parallelschalten der Funktionsgruppen der ursprünglichen Videointerfaceplatine mit denen der Erweiterung, so daß zwischen zwei Arbeitsweisen nur umgeschaltet werden braucht.
Einfacher ist es jedoch, sich nur auf die Erweiterungsfunktion zu beschränken. In diesem Fall wird das EPROM J3 überflüssig am alten Platz. Den 1K-RAM-Bereich kann man separat weiternutzen, indem die Adressenleitungen A0 bis A9 ausschließlich auf den µP-Bus gelegt werden. So kann man auch die Multiplexer J11, J16, J34 als Buffer für das 1K-RAM weiterverwenden, wenn man die Steuereingänge S (Pin 1) auf +5 V legt.
Die nun freie Fassung des EPROM läßt sich gut als Buchse für einen 24poligen DIL-Stecker verwenden,welcher die Verbindung zur Erweiterungskarte darstellt. Lediglich die acht Datenleitungen vom 1K-RAM zur Fassung sowie die vier Versorgungsspannungsleitungen müssen neu belegt werden.
Anstelle von diesen werden die zehn Adressenleitungen für das ehemalige 1K-RAM auf die Fassung gelegt. Außerdem entfallen noch die Bausteine J35 und J30, da eine Positiv-Negativ-Umschaltung in der angegebenen Form nicht mehr nötig ist. Das EPROM J3 läßt sich noch als Zeichengenerator verwenden, nur wird es dann softwareseitig angesteuert zum Datentransfer in den 6K umfassenden Bildwiederholspeicher.

**Schreiben von 67 Hex = 01100111 auf die 1. Speicherstelle (= E000H) bewirkt auf dem Bildschirm folgendes:**

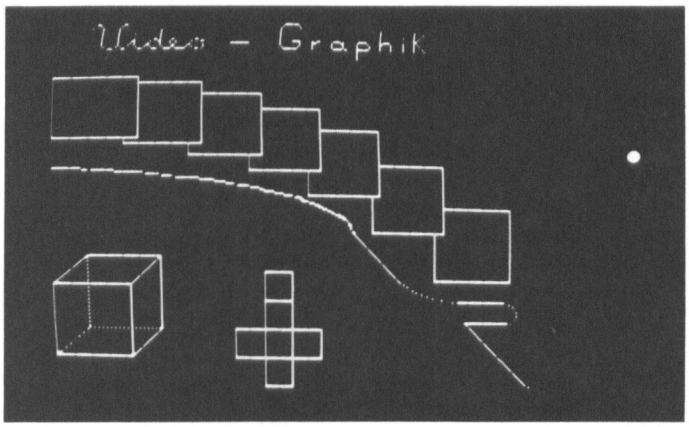



**24polige DIL-Fassung mit neuer Pinbelegung:**



**Aufgebaute Schaltung des Autors auf einer Europakarte**




**Darstellungen der Video-Grafik, wobei die Möglichkeiten durch die punktweise Erfassung gegeben sind**

Ernst Ahlers

# Miniport für TRS-80

Diese Schaltung ist für Verwendung als Eingangs-Port für den TRS-80 Level 2 gedacht.

Da sich der Hauptteil der Schaltung auf vier TTL-Bausteine beschränkt, wurde die Schaltung auf einer Lochrasterplatte aufgebaut. Es empfiehlt sich, beim Nachbau IC-Sockeln zu verwenden.

Die Schaltung dekodiert die „unteren" vier Adressenbits (acht wären möglich, aber 255 Port werden wohl selten benötigt). In dieser Schaltung reagiert das Gerät auf die Port-Adresse 01 hex. (≙ 1 dez.). Durch Hinzufügen bzw. Weglassen von Invertern kann jede Adresse von 0H bis 1EH (≙ 0 bis 14 dez.) eingestellt bzw. adressiert werden.

Die Port-Adresse 0FH kann wegen der nichtdekodierten „oberen" vier Adressenbits nicht verwendet werden (der Miniport würde dann auf FFH ansprechen, welches die Adresse des Kassetten-Ports ist). Im übrigen spricht die Schaltung nicht nur auf die verdrahtete Grundadresse, sondern auch auf die Grundadresse plus alle ganzzahligen Vielfachen von 10H an, d.h. z.B. auf 1FH, 2FH etc. bis FFH an.

Die Einstellung der Adresse erfolgt an den vier Eingängen des UND-Gatters U1, z.B.:

A0,1,2 nicht negiert; A3 negiert – – – Adresse 07H ≙ 7D.

A0,3 nicht negiert; A1,2 negiert – – – Adresse 09H ≙ 9D.

Die vier Adressenleitungen gelangen auf die Eingänge von U1, dessen Ausgang nur zu H-Pegel wird, wenn (in dieser Schaltung) A0 auf „H" und A1-3 auf „L" sind (Port-Adresse 01H).

Der Ausgang des zweiten UND-Gatters U2 geht erst dann auf „H", wenn:

1-A0 auf „H"/A1-3 auf „L" und
2-IN* auf „L" sind.

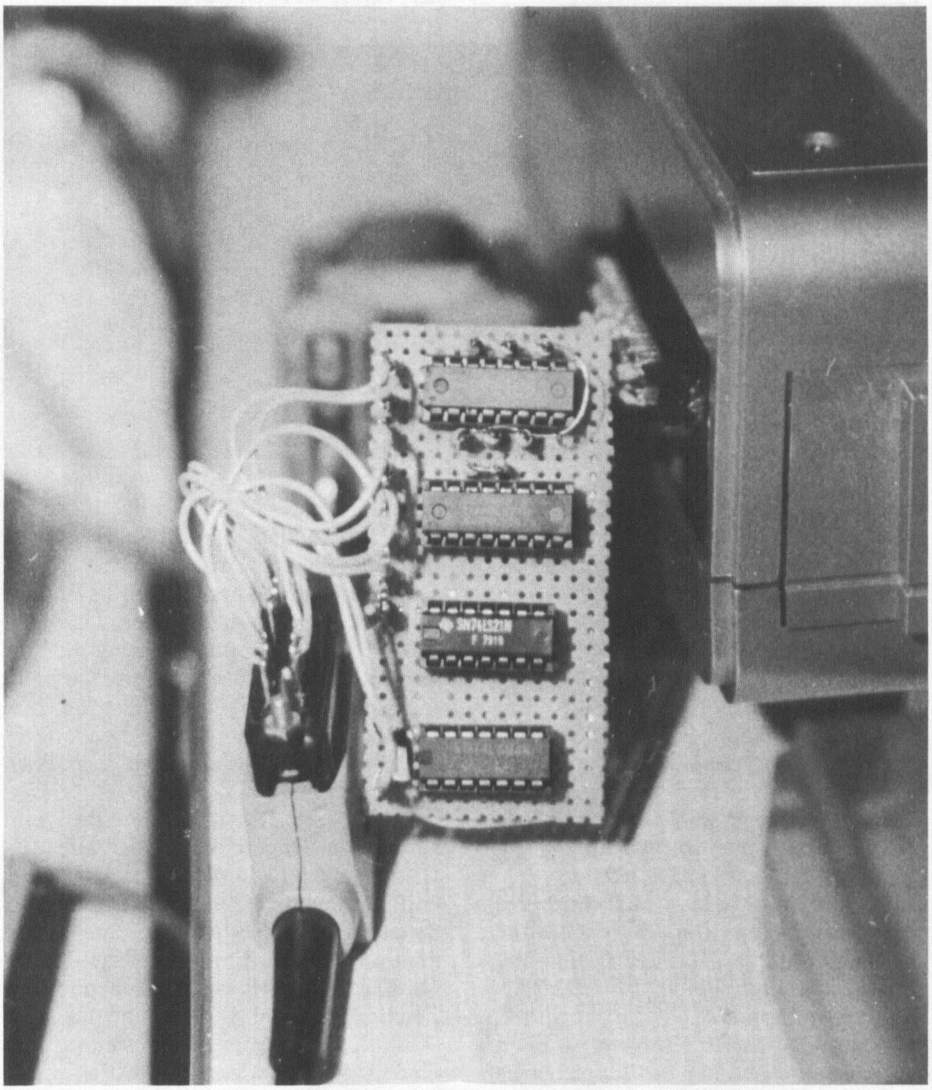
Der IN*-Ausgang des TRS-80 geht nur dann auf „L", wenn durch einen Assemblerbefehl ein Port abgefragt wird (in BASIC entspr.: INP(XX)).

Sind beide Bedingungen erfüllt, geht der Ausgang von U2 auf „H" und über den Inverter N5 der Kontrolleingang der Buffer-Bausteine auf „L".

Dadurch werden die Buffer aktiviert, und die Bufferausgänge, die an den Datenbus des TRS-80 angeschlossen sind, folgen den Eingangssignalen (E0 – D0, etc.).

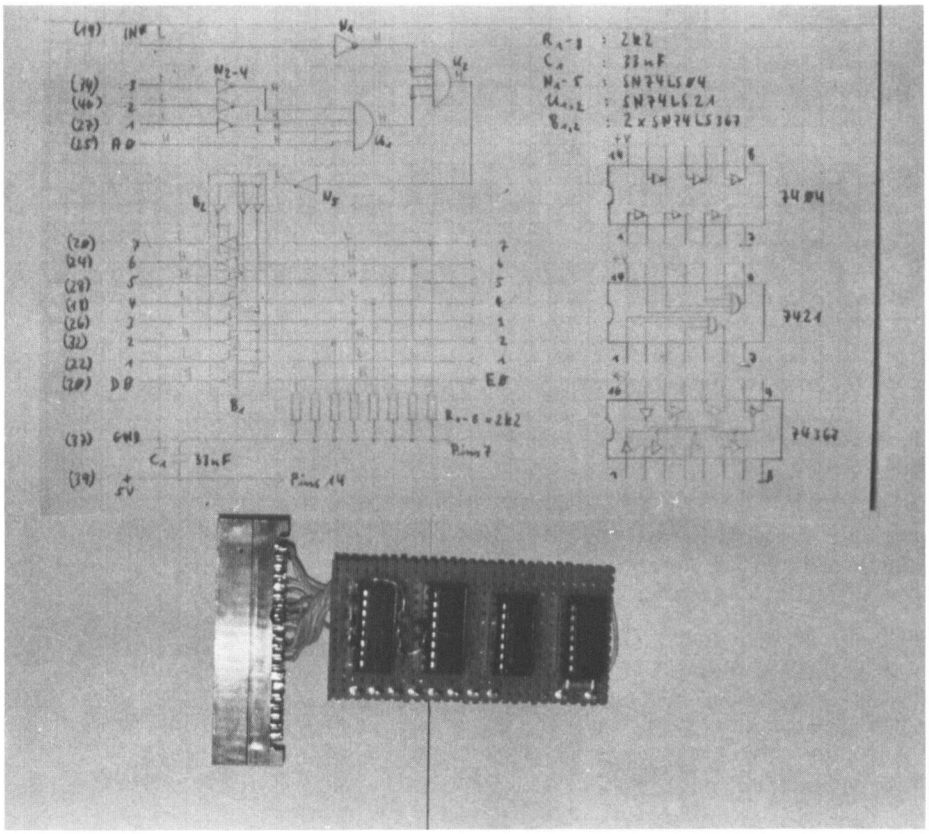
Die R1 – 8 dienen als Pull-Down-Widerstände, um bei unbeschalteten Eingängen einen definierten Zustand zu erzeugen oder in Verbindung mit einpoligen Schaltern (oder Tastern) eine „logical true"-Schaltweise zu erzeugen (Schalter offen = „L", Schalter geschlossen = „H"). Der Kondensator C1 dient als Abblockkondensator.

Die Schaltung kann direkt (über Pin 39) oder durch ein 5-V-Netzteil versorgt werden. Wird das Gerät durch ein Netzteil versorgt, darf Pin (39) nicht angeschlossen werden. Der Anschluß an die Signale des TRS-80 erfolgt über die 40polige „Bus-Card-Edge" des Expansion-Interfaces oder am Expansion-Connector des Keyboards, wenn kein EI vorhanden ist. Die im Schaltbild umklammerten Zahlen sind die Pin-Nummern der Steckkartenecke. Zur Verbindung verwendet man am besten den dafür vorgesehenen Original-Stecker, um eine größtmögliche Kontaktsicherheit zu erhalten. Man kann eventuell auch andere 40polige Steckverbinder benutzen, wenn sie denselben Kontaktabstand haben.

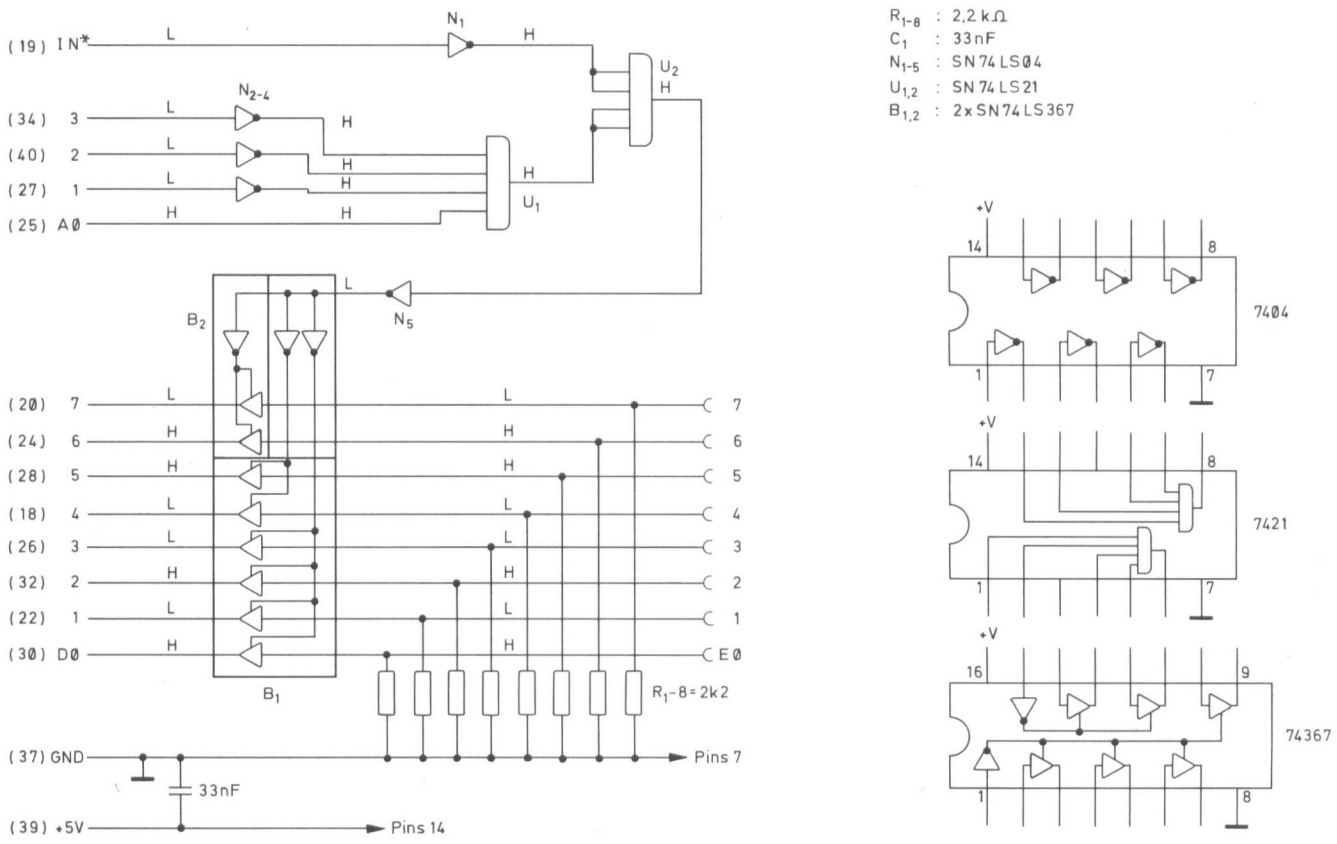


**Miniport, aufgebaut mit Standardelementen, für den TRS-80**

Die Schaltung kann auch durch eine kleine Änderung als Ausgangs-Port verwendet werden. Dazu vertauscht man die Ein- und Ausgänge der Buffer, um die Datenrichtung zu ändern, und statt des IN*-Signals (Pin 19) wird das OUT*-Signal (Pin 12) verwendet. Dann können R1 – 8 weggelassen werden. Allerdings sollte man die Bufferausgänge nicht mit mehr als ca. 10 mA belasten, man kann aber bei Bedarf Treibertransistoren nachschalten.



**Der Miniport für den TRS-80 besteht nur aus vier TTL-Standard-Schaltkreisen und wird auf einer kleinen Lötplatte untergebracht**




**Schaltung des Miniports für den TRS-80 mit gepufferten Datenleitungen und der Adressensteuerung**

Stefan Burger

# Preiswertes Interface zur seriellen Datenausgabe für alle 6502-Mikrocomputersysteme

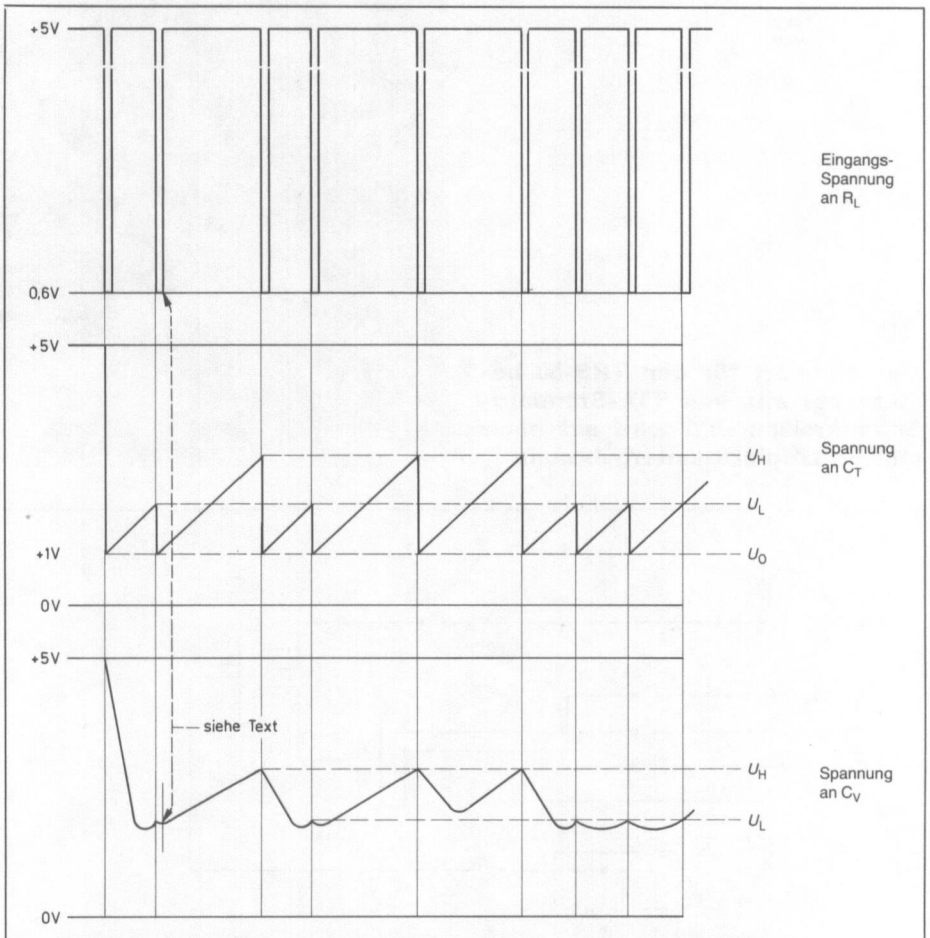
Jeder Mikrocomputerbesitzer wird früher oder später einmal vor dem Problem stehen, wie er digitale Information aus seinem Mikrocomputer an andere Geräte ausgeben kann.

Die Software wird ihm keine Schwierigkeiten bereiten. Kopfzerbrechen macht ihm vielmehr die Hardware.

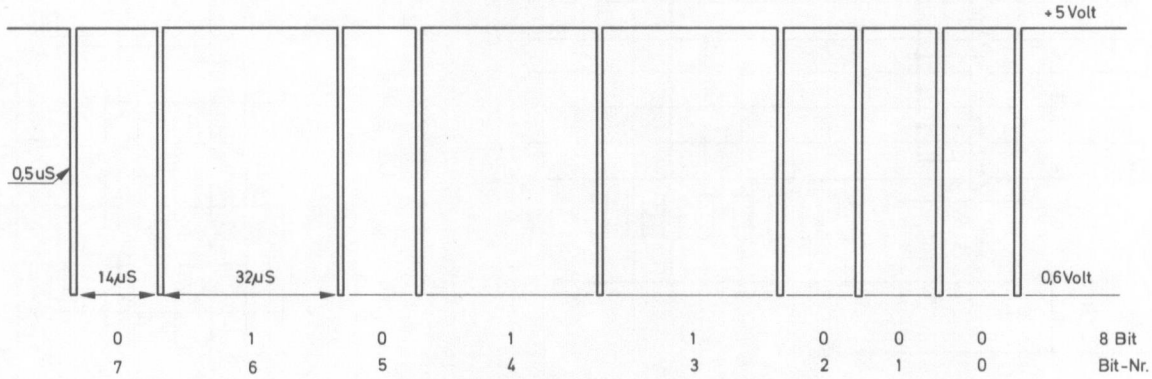
Es sollen Informationen über Schaltzustände oder z.B. auch digitalisierte Spannungswerte übertragen werden. Am wirtschaftlichsten ist dabei für größere Distanzen die serielle Übertragung.

Im folgenden wird ein einfaches, preiswertes und nachbausicheres serielles Interface beschrieben. Über vier getrennte Kanäle kann digitale Information übertragen werden. Sie steht an den Empfängern jeweils als 8-bit-parallele Information zur Ansteuerung von Transistor oder integrierten Schaltkreisen zur Verfügung.

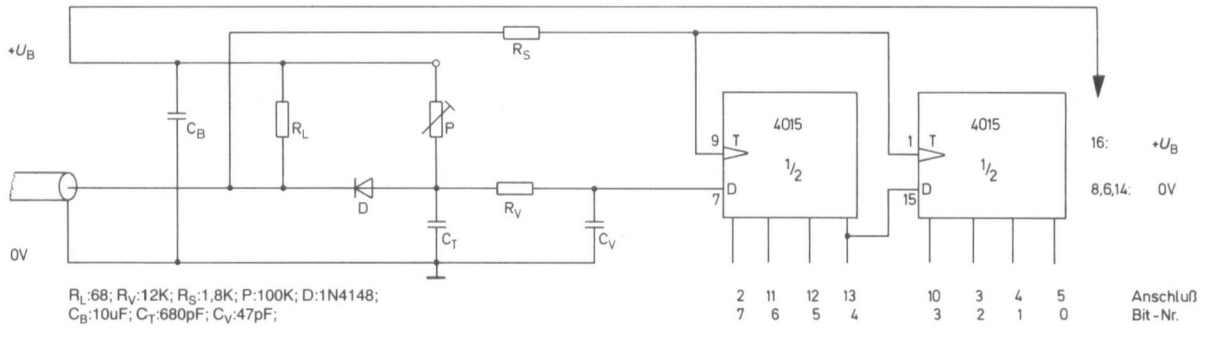
Auf den folgenden Seiten wird die Schaltung und deren Funktion mit der erforderlichen Software erläutert. Es wird das Einstellen des Empfängers mit einem Vielfachmeßinstrument beschrieben, und es sind Schaltungen zur Ansteuerung von Lampen und Motoren, eines Taschenrechners und einer elektronischen Schreibmaschine, einer LED-Großanzeige und ein DA-Wandler aufgeführt.




**Ausgangssignal des Senders bei der Übertragung von 58 Hex**




**Zeitlicher Verlauf der Spannungen in der Schaltung**




**Gesamtschaltung des Empfängers**

## Die Empfängerschaltung und Aufbau des seriellen Kodes

Die serielle Bitinformation wird im Mikrocomputer durch Software (parallele-serielle-Wandlung) erzeugt und über den Sender ausgegeben. Im Empfänger wird sie durch ein Schieberegister in die parallele Form zurückverwandelt. Das Schieberegister (SR) muß dazu ein Taktsignal an T und das Informationssignal an D erhalten.

Takt und Information sind beide im Ausgangssignal des Senders enthalten.

Den Takt stellen die kurzen negativen Impulse dar (500 ns). Die Information ist durch zwei unterschiedliche Pausenzeiten kodiert. L-Pegel entspricht $T_L$ = 14 µs; H-Pegel entspricht $T_H$ = 32 µs. Takt und Information müssen im Empfänger wieder getrennt werden.

In der Schaltung des Empfängers werden die Impulse direkt als Takt für das Schieberegister verwendet. Die Information wird durch ein einfaches RC-Glied zurückgewonnen. Das RC-Glied setzt die zwei Pausenzeiten in zwei verschieden hohe Spannungen um, denen man L und H zuordnen kann.

Der zeitliche Verlauf der Spannungen in der verwendeten Schaltung ist abgebildet. Zur besseren Darstellung sind die Zeitverläufe nicht maßstäblich gezeichnet.

Mit jedem Taktimpuls wird die Kondensatorspannung $U_{CT}$ auf einen Anfangswert $U_0$ gesetzt. Sie hat dann bis zum nächsten Impuls den Wert $U_L$ bzw. $U_H$ erreicht. $U_{CT}$ wird über ein weiteres RC-Glied, welches diese Spannung für kurze Zeit speichern kann, auf den D-Eingang des SR gegeben. Mit dem nächsten Taktsingal wird die Information in das SR übernommen.

Das zweite RC-Glied ist notwendig, da der Taktimpuls innerhalb von 200 ns die Spannung $U_L$ bzw. $U_H$ auf den Anfangswert $U_0$ zurücksetzt. Das SR wird die Information aber erst mit der steigenden Flanke des Taktsignals übernehmen.

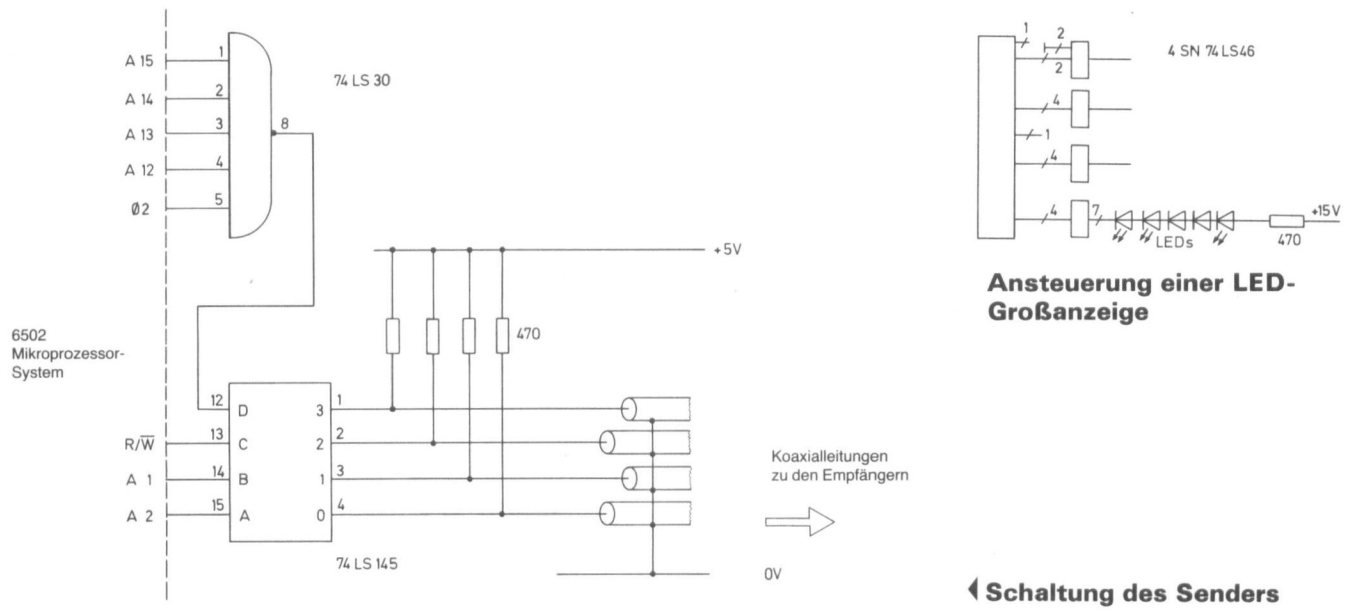
Über die Verbindungsleitung bekommt der Empfänger nicht nur die Information und den Takt, sondern auch die Betriebsspannung für das Schieberegister geliefert.

$R_L$ und $C_B$ bilden einen Tiefpaß, der die Betriebsspannung des SR glättet bzw. die negativen Impulse unterdrückt. $R_L$ ist gleichzeitig der passende Abschlußwiderstand für die Koaxial-Verbindungsleitung, um Reflexionen und dadurch erzeugte Impulsverzerrungen zu vermeiden.

## Die Sender- und Adressendekodierung

Der Sender soll vom Mikroprozessor (µP) gesteuert Impulse abgeben.

Selbstverständlich muß dem Sender zunächst eine Adresse – die Ausgabeadresse – zugeordnet werden. Der µP




**Ansteuerung einer LED-Großanzeige**

**Schaltung des Senders**

# Preiswertes Interface zur seriellen Datenausgabe für alle 6502-Mikrocomputersysteme

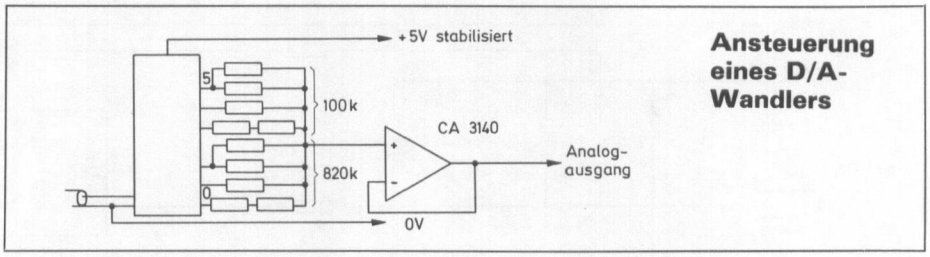
schreibt dann in diese Adresse und erzeugt für die Dauer des Schreibvorgangs (500 ns) einen Impuls an einem Ausgang des Senders. Dabei kommt es nicht darauf an, was der µP in die Adresse schreibt. Es kommt vielmehr nur auf den zeitlichen Abstand zwischen zwei Schreibbefehlen bzw. Schreibimpulsen an.

An die Senderschaltung müssen also keine Leitungen vom Datenbus, sondern nur einige Adressenbit mit 02 zur Adressendekodierung und der R/$\overline{W}$-Strobe angeschlossen werden. Welche Adresse im 64K-Adreßbereich des µP wird jetzt zur Ausgabe benutzt?

Um die beschriebene Schaltung an alle 6502-µP-Systeme anschließen zu können, muß sie in einem Bereich liegen, der mit Sicherheit noch bei keinem System zur Ausgabe benutzt wird, also ein Bereich, in dem der µP noch keine Schreibbefehle ausführt. Der Bereich kann allerdings zum Lesen bzw. zur Eingabe von Daten genutzt sein. Wenn er jetzt zusätzlich zur Ausgabe verwendet wird, kann man das ja eindeutig am R/$\overline{W}$-Strobe erkennen. Bei µP-Systemen mit dem Prozessor 6502 ist mindestens der Bereich F000 bis FFFF Hex (Block F) für Monitor-PROM-Baustein reserviert. Ein fehlerloses Programm wird nie in diesen Bereich schreiben, da der Schreibbefehl für PROM-Bausteine sowieso wirkungslos ist. Die Ausgabeadresse kann in diesen Bereich gelegt werden, da ein Schreibbefehl in Block F nur für den Sender bestimmt sein kann.

Mit den Adressenbit A12 bis A15 wird Block F dekodiert. A0 und A1 wählen die Ausgänge 0 bis 3 an.

Für die Übertragung von 8 bit muß der µP neun Impulse ausgeben, da der Empfänger acht Pausenzeiten messen muß. Für die Information 00H werden 160 µs und für FFH 340 µs CPU-Zeit benötigt. Im Mittel kann man mit 250 µs rechnen. Das sind 0,025% einer Sekunde und fällt auch bei mehreren Ausgaben pro Sekunde und Berücksichtigung des Steuerprogramms kaum ins Gewicht.




**Ansteuerung eines D/A-Wandlers**

## Die Software

Das Interface wird durch ein kurzes Unterprogramm angesteuert. Bevor es aufgerufen wird, muß die 8-bit-Information im Akkumulator und die Nummer des Ausgabekanals im X-Register abgelegt werden. Listing des Programms lautet (siehe untenstehende Tabelle):

Das Unterprogramm wandelt die Information „parallel-seriell" durch einen ROL-(rotate left-)Befehl. Von Bit 7 bis Bit 0 werden in fallender Reihenfolge alle Bit vom Register F0 in der Zeropage (00F0) in das Carryflag geschoben.

Der Befehl BCC (branch on carry clear) in Zeile 7 entscheidet, ob eine lange Pause von 32 µs oder eine kurze von 14 µs zwischen zwei Schreibbefehle STA A,X FFF0 gesetzt wird. Diese Befehle erzeugen die Impulse an den Ausgängen 0 bis 3, wenn der Inhalt des X-Registers 00H bis 03H ist. Die Programmschleife, die die Datenbit aus 00F0 in das Carryflag schiebt und die Pausenzeiten bestimmt, wird so lange durchlaufen, bis kein gesetztes Bit mehr in 00F0 vorhanden ist. Das letzte gesetzte Bit – es wurde durch den Befehl SEC in Zeile 2 erzeugt – verläßt mit dem neunten Schiebebefehl das Register 00F0. Nach dem neunten Schiebebefehl wird mit der Verzweigung BEQ (branch on equal – besser: branch on zero) aus der Schleife herausgesprungen. Die acht Datenbit sind dann übertragen, und das Unterprogramm kommt mit RTS zurück.

In fast allen 6502-Systemen ist bei der Adressendekodierung der Monitor-PROMS der R/$\overline{W}$-Strobe nicht mitdekodiert, da man ja aus diesem Bereich sinnvollerweise nur liest.

Das PROM wird also auch bei einem Schreibbefehl das Datum der angesprochenen Adresse auf den Datenbus legen.

Damit es auf dem Datenbus keine „Kollisionen" gibt, muß der µP dasselbe Datum angeben, das bei der Software in Zeile 3 berücksichtigt ist.

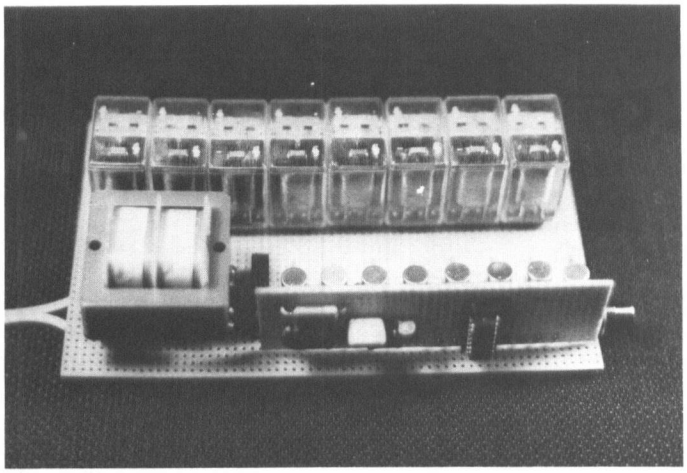
## Das Einstellen des Empfängers

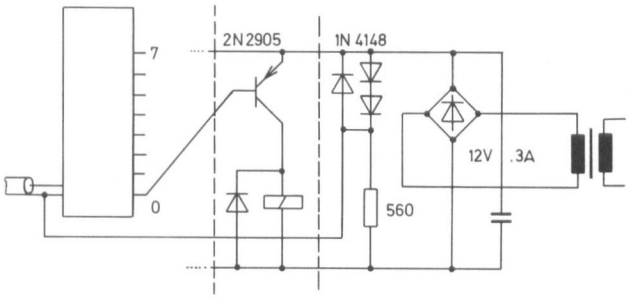
Nachdem die Schaltung fertig aufgebaut ist, muß der Empfänger noch auf „optimalen Empfang" eingestellt werden. Mit dem Trimmer P soll die Grenze für die Unterscheidung zwischen high und low auf 21,5 µs = $T_G$ eingestellt werden. Dadurch ist das Verhältnis von $T_H$ zu $T_G$ etwa gleich $T_G$ zu $T_L$.

Das Programm erzeugt zwei Impulse mit einer Pause, die abwechselnd 20 und 23 µs beträgt. Liegt die Zeitgrenze beim Empfänger zwischen diesen beiden Werten, so wechselt z.B. der Ausgang Nr. 0 – Pin 5 – mit der Übertragung jedes Impulspaares seinen Zustand.

Um den Empfänger mit Hilfe eines Viel-

```
Zeile                                    A  B  C  **
 1            STA-Z-F0        85 F0              3  Inf. in Zeropage ablegen
 2            SEC             38                 2  für Schleifenzähler
 3            LDA A,X FFF0    BD F0 FF           4  siehe*) in 3)
 4 Schleife   STA A,X FFF0    9D F0 FF  4  4        Impuls auf Kanal (X)
 5            ROL-Z-F0        26 F0     5  5        Inf.-Bit in Carry schieben
 6            BEQ 'Ende       F0 09     2  2  3     letztes Bit aus Zeile 2 draußen?
 7            BCC 'Schleife   90 F7     3  2        1 oder 0? Bei 0 keine zusätzl. Verzögerung
 8            PHA             48                 3  Zeitverzögerung
 9            PLA             68                 4  Zeitverzögerung
10            PHA             48                 3  Zeitverzögerung
11            PLA             68                 4  Zeitverzögerung
12            CLC             18                 2  clear Carry für Zeile 5
13            BCC 'Schleife   90 F0              3  unbedingter Sprung
14 Ende       RTS             60                 6  fertig!

** Zeitbedarf in µs bei 1 MHz-Takt: A) kurze Pause; B) lange Pause; C) vor/nach Ausgabe
```






**Schaltung zur Ansteuerung eines Relais**

**Ansicht einer aufgebauten Schaltung**

fachmeßinstruments einzustellen, geht man folgendermaßen vor:

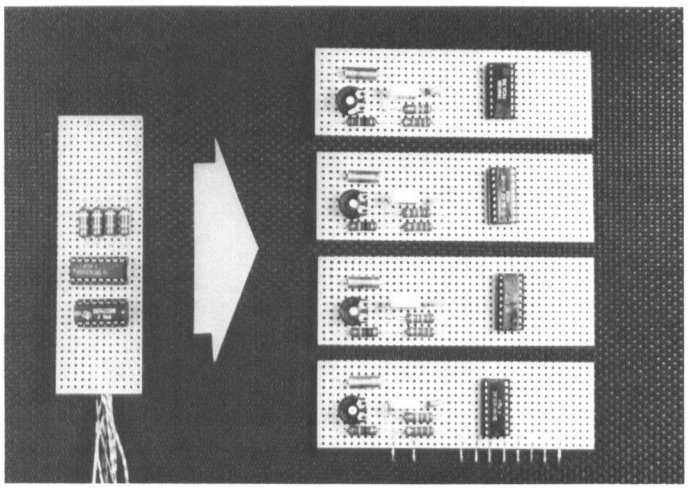
- P wird in Maximalstellung gebracht.
- Ein Vielfachmeßinstrument wird zwischen die Pins 5 und 16 des Schieberegisters angeschlossen.
- Für den Befehl LDX imm ?? wird je nach Nummer des Ausgabekanals LDX 00 bis LDX 03 eingesetzt (A2 00 bis 03).
- Das Hilfsprogramm wird gestartet.
- Der Trimmer P wird langsam zu kleineren Werten hin verändert.

Das Meßinstrument zeigt zunächst $+U_b$, die Betriebsspannung des Schieberegisters, an und geht dann auf 0V zurück. Bei der Einstellung des Trimmpotentiometers gibt es einen kleinen Bereich zwischen 0V und $+U_b$, bei der die Anzeige um den Mittelwert $+U_b/2$ schwankt.

Der Empfänger ist dann richtig eingeregelt, wenn diese Einstellung gefunden ist.

Das Programm lautet:

**Aufbau der einzelnen Platinen zu einem freiprogrammierten Interface**



## Schaltungsvorschläge

**Ansteuerung von 220-V-Verbrauchern.** Die acht Ausgänge des Empfängers steuern über Transistoren acht Relais an.

**Ansteuerung eines Taschenrechners oder einer elektronischen Schreibmaschine.** Prinzip: Die Tastaturmatrix wird durch CMOS-Analogschalter simuliert. Um die Taste zu betätigen, werden deren „Koordinaten" übertragen – um sie wieder loszulassen, werden die Koordinaten einer Tastenreihe oder Spalte übertragen, die nicht angeschlossen ist. Andere Möglichkeit: Bei der Übertragung wird ein Monoflop angestoßen, der über die Enable-Eingänge der Analogschalter die entsprechende Taste für z.B. 100 ms betätigt. Bei der Verwendung von CMOS-Schaltkreisen kann die Schaltung vom µP aus mit Strom versorgt werden.

**Ansteuerung einer LED-Großanzeige** – z.B. einer Uhr.

Die Segmente der Anzeige werden durch bis zu fünf in Serie geschaltete LED dargestellt. Für vier Stellen werden bei einer Uhr mindestens 14 bit benötigt. Deshalb wird auf der Empfängerplatine noch ein zusätzliches Schieberegister angeschlossen.

**D/A-Wandler.** Bei CMOS-Schaltkreisen kann ein hochohmiges Widerstandsnetzwerk direkt an die Ausgänge angeschlossen werden. Der Empfänger muß an eine gut stabilisierte Spannungsquelle angeschlossen werden.

```
 1 Start      LDX imm ??     A2  ??          Nr. des Kanals
 2 Endlos     LDA A,X FFF0   BD  F0  FF
 3            STA A,X FFF0   9D  F0  FF   4  Impuls auf Kanal X
 4            PHA            48          3  Pause 20 µs
 5            PLA            68          4
 6            PHA            48          3
 7            PLA            68          4
 8            NOP            EA          2
 9            STA A,X FFF0   9D  F0  FF      Impuls
10 Verz. 1    DEY            88             Warteschleife 1 ms
11            BNE 'Verz. 1   D0  FD
12            STA A,X FFF0   9D  F0  FF   4  Impuls auf Kanal X
13            PHA            48          3  Pause 23 µs
14            PLA            68          4
15            PHA            48          3
16            PLA            68          4
17            SEC            38          2
18            BCS ---        B0  00      3  springt in Zeile 19
19            STA A,X FFF0   9D  F0  FF      Impuls
20 Verz. 2    DEY            88             Warteschleife 1 ms
21            BNE 'Verz. 2   D0  FD
22            BEQ 'Endlos    F0  DD          unbedingter Sprung
```

Frank Helbig

# AIM-65 als interrupt-gesteuerter Frequenzzähler

In jüngster Zeit wurden die verschiedensten Bauanleitungen für Induktivitäts- und Kapazitätsmesser vorgestellt, die alle nach dem gleichen Prinzip aufgebaut sind: Im Meßteil der Schaltung wird entsprechend der jeweils angelegten Induktivität oder Kapazität eine Frequenz erzeugt, die anschließend in einer relativ aufwendigen Schaltung zur Anzeige gebracht wird.
Daraus entstand der Wunsch, die jeweilige Frequenz extern quarzgenau zu messen, das Ergebnis in den Mikrocomputer einzulesen und anschließend softwaremäßig in die gewünschten Kapazitäts- bzw. Induktivitätswerte umzurechnen.

## Die Schaltung

Die aus zwei Teilen bestehende Schaltung wurde so einfach wie möglich gestaltet und die Steuerung auf die Software-Seite verlegt.
Der erste Teil der Schaltung besteht aus acht Dekaden-Zählern 7490, deren BCD-Ausgänge auf zwei 16-bit-Multiplexer geschaltet sind. Die Steuerung der Multiplexer erfolgt durch den 4-bit-Zähler am Eingang, der beim Hochzählen Eingang für Eingang zum Ausgang entsprechend durchschaltet.
Der zweite Teil der Schaltung stellt die Quarzzeitbasis dar, die etwas aufwendiger gestaltet wurde, um sie für verschiedene Anwendungsfälle einsetzen zu können. Herzstück dieser Schaltung ist der 10-MHz-Quarzoszillator mit dem MOS-Teiler-IC MK 5009. Um Meßzyklen von 0,01 s, 0,1 s, 1 s usw. zu erreichen, muß die Ausgangsfrequenz nochmals durch zwei geteilt werden.

## Die Programmerläuterungen

Mit den BASIC-Befehlen 10 – 40 wird der Interrupt-Vektor auf die Startadresse des Maschinenprogramms der Interrupt-Routine (OF50 in Speicherplatz A404, A405) und die Startadresse des Maschinenprogramms, das zur Vorbereitung der Interrupt-Routine benötigt wird (OF00 in Speicherplatz 0004, 0005), eingegeben.
Befehl 60 – 90 errechnet aus der eingegebenen Meßzeit (realistische Werte: 0,01; 0,1; 1; 10; 100) den Wert zum Einstellen der Zeitbasis und speichert ihn in Speicherzellen OFFC ab. Durch Ersatz dieser Befehle durch einen POKE-Befehl kann die Zeitbasis auch noch mit anderen Teilungsverhältnissen geladen werden. Mit X = USR (0) wird anschließend das Maschinenprogramm zur Vorbereitung der Interrupt-Routine gestartet. Folgendes wird durchgeführt:

- Verhinderung einer möglichen Interrupt-Routine
- Einstellen des entsprechenden Datenrichtungsregister B
- Reset für Zähler und Zeitbasis
- Einstellung der Zeitbasis mit Speicherstelle OFFC
- Laden des peripheren Kontroll-Registers als negativer Flankendetektor am Anschluß CA2
- Laden des Interrupt-Enable-Registers
- Löschen des Bit 0 im Interrupt-Flag-Register mit anschließendem Rücksprung ins BASIC-Programm.

Mit Befehl 120 – 140 werden die Speicherstellen OFFO – OFF7 gelesen (darin ist der Inhalt der Zählerwerte) und den A(1)- bis A(8)-Variablen unter Berücksichtigung der Meßzeit und Stellenwertigkeit zugeordnet.
Anschließend werden die A(X)-Werte aufaddiert und das Ergebnis angezeigt bzw. ausgedruckt. Indem man anschließend der Meßschaltung entsprechende mathematische Umrechnungsformeln eingibt, kann man das Ergebnis auch sofort als Induktivitäts- oder Kapazitätswert ausdrucken lassen.

## Die Interrupt-Routine

Ist ein Meßzyklus beendet und springt am CA2-Anschluß (Tor-Zeit) das Potential von 1 auf 0, wird das laufende BASIC-Programm unterbrochen und die Interrupt-Routine durchgeführt:

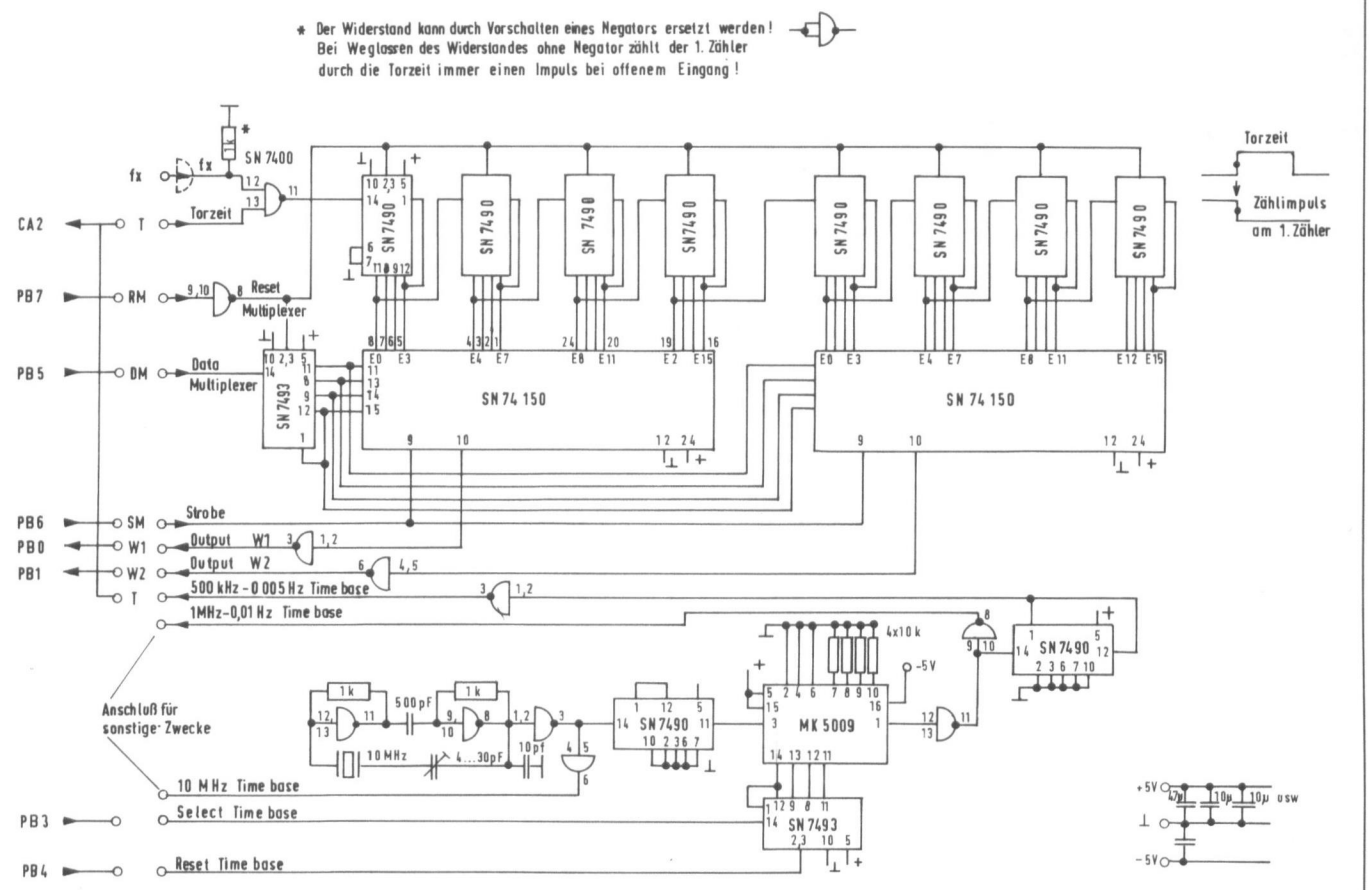
- Retten der Registerwerte
- Auslesen der Zählerwerte von den Dekadenzählern in die Speicherstellen OFF0 – OFF7
- Erhöhen der abfragbaren Speicherstelle OFFA um 1 mit anschließendem Rücksprung ins BASIC-Programm.

```
LIST
 10 POKE41988,80
 20 POKE41989,15
 30 POKE04,0
 40 POKE05,15
 60 INPUT"ZEIT FUER
FREQUENZM";A1
 70 A2=LOG(A1*10000)
/2.3
 80 A2=INT(A2)+2
 90 POKE4092,A2
 100 X=USR(0)
 105 B1=PEEK(4090)
 110 POKE4093,1
 120 FORI=1TO8
 130 A(I)=PEEK(4079+
I)*10^(14-I-A2)
 140 NEXTI
 160 S=0
 170 FORI=1TO8
 180 S=S+A(I)
 190 NEXTI
 195 POKE4093,0
 200 PRINTS
 210 B2=PEEK(4090)
 220 IFB2=B1GOTO210
 230 GOTO105
```

```
<K>*=0F00
/35
 0F00 A9 LDA #01
 0F02 8D STA A00E
 0F05 A9 LDA #F8
 0F07 8D STA A002
 0F0A A9 LDA #50
 0F0C 8D STA A000
 0F0F A9 LDA #C0
 0F11 8D STA A000
 0F14 AC LDY 0FFC
 0F17 F0 BE0 0F26
 0F19 A9 LDA #C8
 0F1B 8D STA A000
 0F1E A9 LDA #C0
 0F20 8D STA A000
 0F23 88 DEY
 0F24 D0 BNE 0F19
 0F26 A9 LDA #02
 0F28 8D STA A00C
 0F2B A9 LDA #81
 0F2D 8D STA A00E
 0F30 A9 LDA #01
 0F32 8D STA A00D
 0F35 60 RTS
<K>*=0F50
/20
 0F50 48 PHA
 0F51 8A TXA
 0F52 48 PHA
 0F53 98 TYA
 0F54 48 PHA
 0F55 AD LDA 0FFD
 0F58 D0 BNE 0F5D
 0F5A 4C JMP 0F80
 0F5D A9 LDA #40
 0F5F 8D STA A000
 0F62 A9 LDA #C0
 0F64 8D STA A000
 0F67 A9 LDA #01
 0F69 8D STA A00D
 0F6C 68 PLA
 0F6D A8 TAY
 0F6E 68 PLA
 0F6F AA TAX
 0F70 68 PLA
 0F71 40 RTI
```

```
<K>*=0F80
40
 0F80 A0 LDY #07
 0F82 A9 LDA #00
 0F84 99 STA 0FF0,Y
 0F87 88 DEY
 0F88 10 BPL 0F84
 0F8A A2 LDX #04
 0F8C A0 LDY #04
 0F8E A9 LDA #80
 0F90 8D STA A000
 0F93 AD LDA A000
 0F96 29 AND #01
 0F98 5D EOR 0FF3,X
 0F9B 0A ASL A
 0F9C 9D STA 0FF3,X
 0F9F AD LDA A000
 0FA2 29 AND #02
 0FA4 5D EOR 0FEF,X
 0FA7 0A ASL A
 0FA8 9D STA 0FEF,X
 0FAB A9 LDA #C0
 0FAD 8D STA A000
 0FB0 A9 LDA #E0
 0FB2 8D STA A000
 0FB5 A9 LDA #C0
 0FB7 8D STA A000
 0FBA 88 DEY
 0FBB D0 BNE 0F8E
 0FBD BD LDA 0FF3,X
 0FC0 4A LSR A
 0FC1 9D STA 0FF3,X
 0FC4 BD LDA 0FEF,X
 0FC7 4A LSR A
 0FC8 4A LSR A
 0FC9 9D STA 0FEF,X
 0FCC CA DEX
 0FCD D0 BNE 0F8C
 0FCF AD LDA 0FFA
 0FD2 69 ADC #01
 0FD4 8D STA 0FFA
 0FD7 4C JMP 0F5D
```

**BASIC- und Maschinenprogramm für die Steuerung**




**Schaltung für den Frequenzzähler**

## Die Synchronisation

Die Synchronisation zwischen BASIC-Programm und Interrupt-Routine wird mit den Befehlen 90, 105, 110, 195, 210, 220 und 230 erreicht.
Bei langen Meßzyklen wartet das BASIC-Programm auf den nächsten Meßwert durch die Interrupt-Routine. Bei kurzen Meßzyklen werden bei entsprechenden Interrupt-Anforderungen nur die Dekadenzähler zurückgestellt, aber keine neuen Werte in die Speicherstellen OFF0 – OFF7 geschrieben (keine verfälschten Werte beim Auslesen).

| OFFC | Periodendauer 1-MHz-Ausgang | Periodendauer 500-kHz-Ausgang |
|---|---|---|
| 0 | 1 µs | 2 µs |
| 1 | 10 µs | 20 µs |
| 2 | 100 µs | 200 µs |
| 3 | 1 ms | 2 ms |
| 4 | 10 ms | 20 ms |
| 5 | 100 ms | 200 ms |
| 6 | 1 s | 2 s |
| 7 | 10 s | 20 s |
| 8 | 100 s | 200 s |
| 9 | 60 s | 120 s |
| A | 3600 s | 7200 s |
| B | 600 s | 1200 s |
| E | 20 ms | 40 ms |

## Die Zeitbasis

Durch Laden der Zeitbasis mit den nachfolgenden Werten (Speicherzelle OFFC) können folgende Teilungsverhältnisse an den beiden Ausgängen eingestellt werden (s. nebenstehende Tabelle).

Mit Hilfe dieser universellen Zeitbasis läßt sich auch leicht eine einfache software-gesteuerte interne Uhr programmieren.

## Der Aufbau

Beim Aufbau wird empfohlen, die beiden Multiplexer-Bausteine SN 74150 mit dem Zählerbaustein SN 7493 und den Teilerbaustein MK 5009 mit dem zugehörigen Steuerbaustein SN 7493 zu sokkeln, um ggf. Fehler in der Schaltung durch Lötspritzer usw. schnell zu eliminieren. Durch Ersatz des Befehls 110 durch POKE 4093,0 (Aufheben der Synchronisation) und Einstellen eines kurzen Meßzyklusses (0,01) läßt sich die genaue Funktionsweise der Schaltung mit Hilfe eines Oszillographen überprüfen.
Speziell beim Teilerbaustein MK 5009 gab es Probleme. Sollte der Teiler nicht ordnungsgemäß arbeiten, so ist zu prüfen, ob alle Ausgänge am SN 7493 über 4,0 V liegen, da bei dem hier verwendeten MK 5009 der Teiler bei 3,8 V nicht zum Umschalten zu bewegen war (ggf. kann auch ein Dekadenzähler SN 7490 verwendet werden; ein Rest-Eingang muß in diesem Fall noch auf Masse gelegt werden).
Das Maschinenprogramm wurde bei dem mit 4 KByte ausgestatteten AIM 65 an das Ende des Speicherbereiches gelegt. Bei einer evtl. Speicherplatzverschiebung müßten die entsprechenden Sprungbefehle abgeändert werden. Da man die beiden Programmteile zweckmäßigerweise auf einen Kassettenrecorder abspeichert, muß man beim Aktivieren des BASIC-Interpreters darauf achten, das geladene Maschinenprogramm nicht wieder zu löschen.

Erich Blahout

# Software-unabhängiger EPROM-Programmer

In letzter Zeit wurden bereits öfters EPROM-Programmer zum Selbstbau veröffentlicht, die aber alle einen gemeinsamen Nachteil hatten: Sie mußten alle an ein bereits vorhandenes Mikrocomputersystem angeschlossen werden. Das hier beschriebene Gerät ist ein kompletter EPROM-Programmer, der zwei Spannungen von +5 V und +25 V benötigt. Das Eingeben der Daten erfolgt über eine auf der Platine vorhandene Tastatur. Der Programmiervorgang wird durch Drücken der PROG-Taste ausgelöst. Dadurch können auch einzelne Daten in einen EPROM eingegeben werden bzw. können EPROM zum Teil korrigiert werden (sofern im gespeicherten Wort ein H-Bit gegen ein L-Bit ersetzt werden soll).

Weiter sind alle Adressen- und Datenleitungen mit Tri-State-Buffern versehen, so daß das Gerät auch als Statusanzeige im Mikrocomputer verwendet werden kann. Mit Hilfe einer Einzelschrittschaltung kann das Funktionieren der entworfenen Programme überprüft werden.

## Die Schaltungsbeschreibung

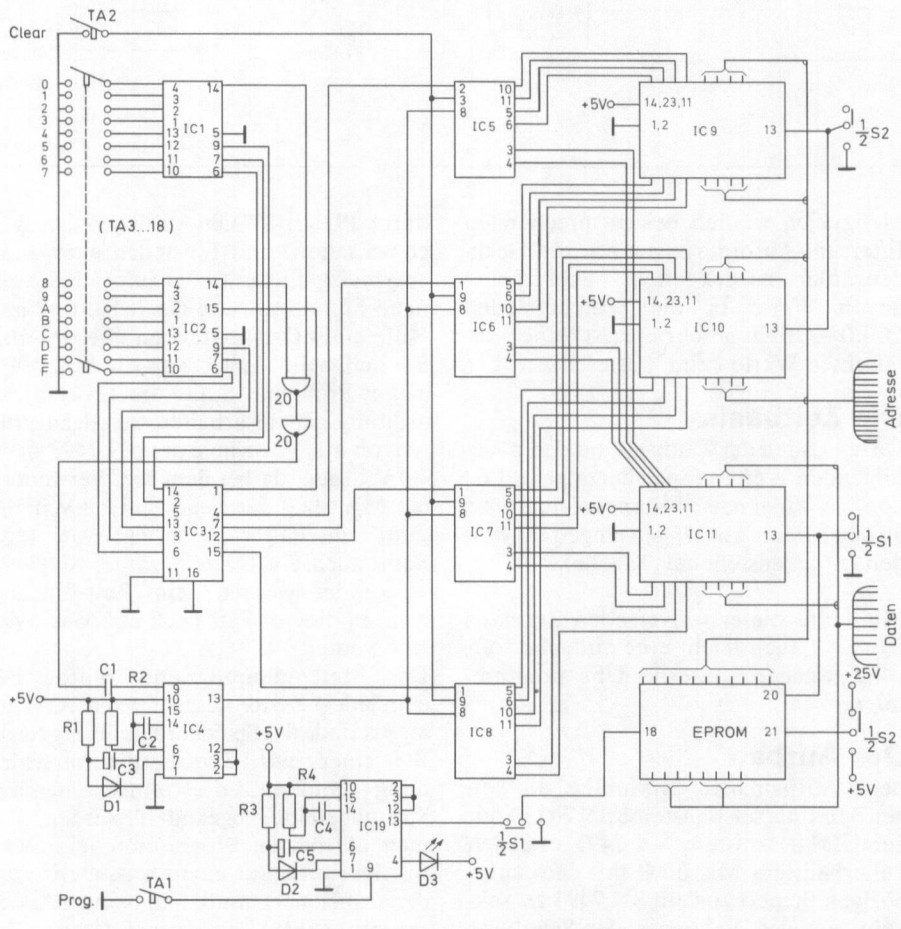
In der Grundstufe ist der Programmer für 2716 EPROM ausgelegt. Mit einem Umschalter lassen sich auch 2732, mit einer Zusatzschaltung auch 2708 programmieren.

Das Gerät besteht im wesentlichen aus drei Blöcken:
- Tastatur/Eingabeeinheit
- Programmer
- Anzeige.

Sofern bereits eine Anzeige vorhanden ist, kann natürlich diese verwendet werden.

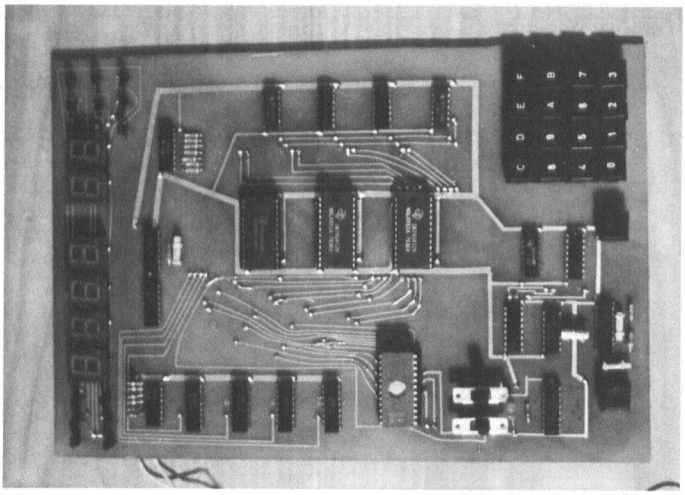
## Die Tastatur und Eingabeeinheit

Beim Drücken einer der Tasten „O" bis „F" wird an einen der Eingänge von IC1 oder IC2 (zwei 3-bit-Dezimal-zu-Binärwandler) ein L-Pegel gelegt. An den Ausgängen von IC1(2) erscheint dadurch das entsprechende Binärwort. Da aber nur drei Bit umgewandelt werden, wird in IC3 (zweifach Multiplexer) eine Umkodierung vorgenommen, so daß an den Ausgängen von IC3 das entsprechende Binärwort erscheint.

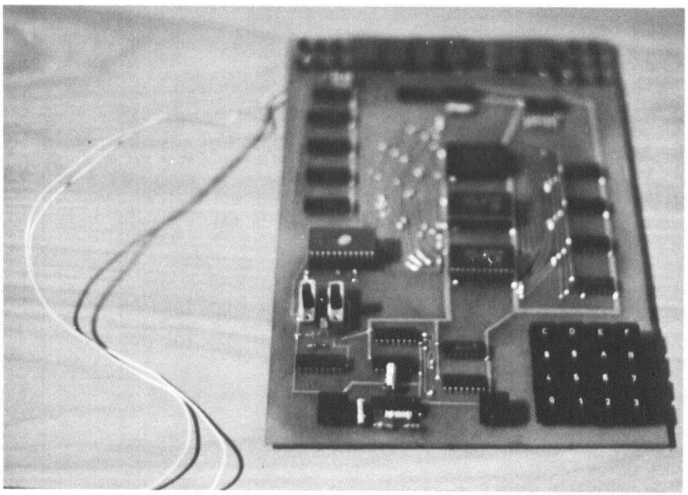



**Tastatur-, Eingabe- und Programmiermodul**

| | |
|---|---|
| IC 1, 2: | 74 148 |
| IC 3: | 74 157 |
| IC 4, 19: | 74 123 |
| IC 5, 6, 7, 8: | 74 164 |
| IC 9, 10, 11: | 74 412 (8212) |
| IC 12—15: | 74 151 |
| IC 16: | 9368 |
| IC 17: | 7413 |
| IC 18: | 7493 |
| IC 20: | 7400 |
| IC 21: | 7442 |
| T1 ··· T7: | BC 237 oä |
| T8 ··· T13: | BC 307 oä |
| D1, D2: | 1 N 4001 oä |
| D3: | Lumineszenzdiode |
| A1 ··· A6: | FND 507 |
| R1, 2, 3: | 47 kΩ |
| R4: | (100 kΩ) |
| R5 ··· R11: | 47 kΩ |
| R12: | 330 Ω |
| R13 ··· R18: | 10 kΩ |
| C1, C2: | 1 nF |
| C3: | 1 µF |
| C4: | (470 nF) |
| C5: | 33 µF |
| C6: | 470 nF |
| TA1 ··· TA18: | Printtaster |
| S1, S2: | Schiebeumschalter |



**Aufbau eines Hardware-EPROM-Programmer**



**Rechts die Tastatur des EPROM-Programmer**

Dieses Binärwort wird in vier Schieberegistern (IC5 bis 8) gespeichert bzw. an die richtige Stelle geschoben. (Die zuletzt eingegebene Ziffer erscheint immer ganz rechts, alle anderen werden um eine Stelle nach links verschoben – wie im Taschenrechner.)

Das Weiterschalten der Schieberegister wird durch IC4 ausgelöst: Beim Drücken irgendeiner Taste wird am Pin 14 von IC1(2) ein L-Impuls abgegeben. Dieser L-Impuls wird in IC20 verknüpft und an den Eingang des Monoflops IC4 gelegt. Im ersten dieser Monoflops erfolgt nur eine Zeitverzögerung, damit ein eventuelles Prellen der Tasten ohne Wirkung bleibt. Sollten die Tasten dennoch prellen, braucht nur C3 vergrößert werden. Der Ausgangsimpuls von IC4 schaltet dann die Schieberegister weiter.

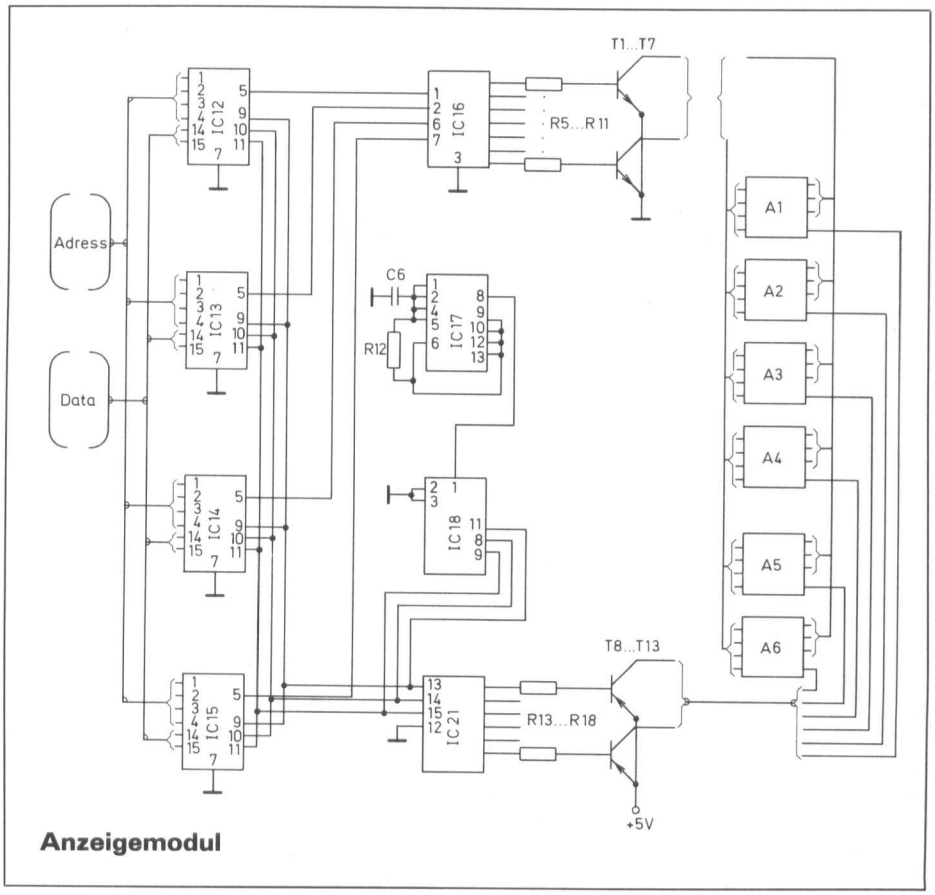
Weiter ist an Pin 9 der Schieberegister eine CLEAR-Taste angeschlossen, die die gesamte Anzeige auf 0 stellt.

An den Ausgängen der IC 5 bis 8 erscheinen die eingegebenen Daten stellenrichtig. Dabei ist zu beachten, daß jeweils dasselbe Pin aller Bausteine ein Wort ergibt (z.B. alle Pin 3 ergeben die niederwertigste Stelle). Diese Daten werden anschließend durch die Bausteine 9, 10, 11 gepuffert. Diese Bausteine haben einen Tri-State-Ausgang, dadurch können im Anschluß an eine erfolgte Programmierung die Daten zur Kontrolle gelesen werden, was sehr praktisch ist.

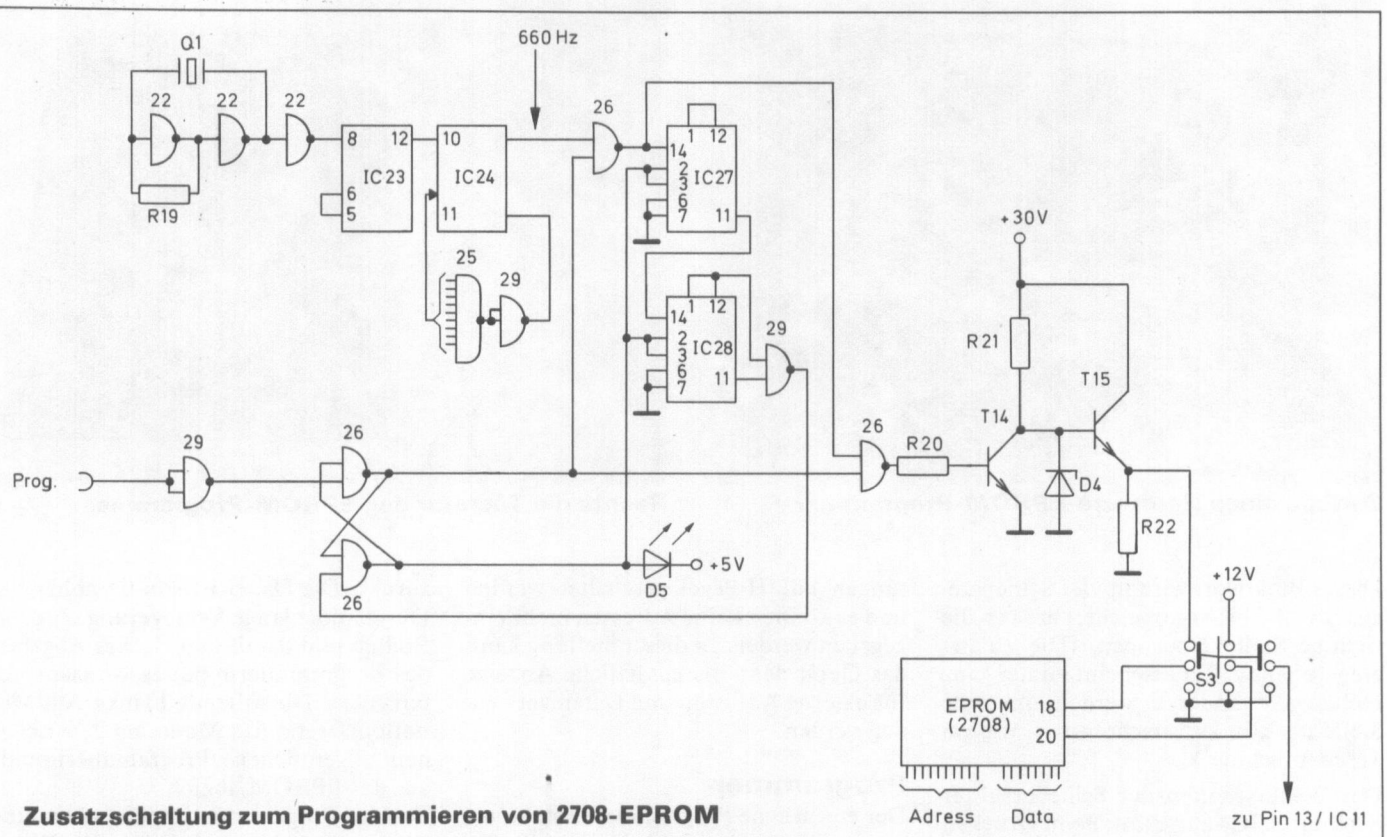
Das Umschalten von Programmier- auf Lesebetrieb erfolgt durch Umschalten von S1. Ist S1 in Richtung der Masseleitung geschaltet, erfolgt das Auslesen der Daten. Bei noch nicht programmierten EPROM erscheint auf der Anzeige FF. Ein Programmieren ist in dieser Stellung nicht möglich, da der PROG-Eingang der 2716 auf L-Pegel gehalten wird.
Mittels S2 können auch die Adressenleitungen auf H-Pegel geschaltet werden und es können keine Adressen mehr eingegeben werden. In dieser Stellung kann das Gerät dann als zusätzliche Anzeige für externe Adressen und Daten verwendet werden.

## Programmer

Der eigentliche Programmer besteht nur aus dem Baustein IC 74123. Beim Drükken der Taste „PROG" wird an das erste Monoflop des IC19 ein L-Signal gelegt. Dadurch wird dieses Monoflop gesetzt und fällt nach etwa einer Sekunde wieder zurück. Die Dauer ist von C5 abhängig. Durch diese lange Verzögerung wird ein Prellen und damit mehrfaches Abgeben des Programmierimpulses wirksam unterdrückt. Die fallende Flanke von Monoflop 1 setzt das Monoflop 2, welches den eigentlichen Programmierimpuls auf das EPROM abgibt.
Dieser Programmierimpuls soll eine Länge von 50 bis 55 ms haben. Die Werte von C4 und R4, welche die Länge dieses Impulses bestimmen, sollten nur als Richtwerte betrachtet werden und sind deshalb in Klammer gesetzt. Man sollte




**Anzeigemodul**




**Zusatzschaltung zum Programmieren von 2708-EPROM**

| | | | | | |
|---|---|---|---|---|---|
| IC 22, 26, 29: | 7400 | T14, T15: | BC 337 | R21: | 10 kΩ |
| IC 23: | 74196 | | | R22: | 47 kΩ |
| IC 24: | 4020 (CMOS!) | D4: | ZD 27 oä | | |
| IC 25: | 7430 | D5: | Lumineszenzdiode | Q1: | Quarz 1 ··· 30 MHz |
| IC 27, 28: | 7490 | | | | |
| | | R19: | 100 Ω | S3: | Schiebeumschalter |
| | | R20: | 3,3 kΩ | | |

unbedingt die Länge des Impulses messen und auf ca. 53 ms einstellen. Ist der Impuls zu kurz, wird das EPROM nicht vollständig und vor allem nicht dauerhaft programmiert. Ist der Impuls zu lang, kann der Baustein zerstört werden. Ein Oszilloskop ist erforderlich.
Gleichzeitig mit der Programmierung leuchtet die LED D3 auf und zeigt somit den Programmiervorgang an.
Durch diese Reduzierung des Programmers auf nur einen einzigen Baustein muß noch auf ein wichtiges Detail hingewiesen werden: Beim Anstecken des Gerätes an die Spannungsversorgung kommt es unweigerlich zu einem Auslösen eines Impulses. Da zu diesem Zeitpunkt die Adressen und Daten nicht vorhersagbar sind, müssen sowohl S1 als auch S2 in der H-Position sein. Dadurch ist ein Fehlprogrammieren sicher verhindert. Anschließend muß zuerst S1, dann erst S2 in die Programmierposition gestellt werden. Ebenso sollten die beiden Schalter sich vor dem Abschalten wieder in der H-Position befinden.
Ein Ein- und Ausstecken des EPROM bei angelegter Spannung ist nicht zu empfehlen!

## Die Anzeige

Die Anzeige der Adressen und Daten erfolgt mittels sechsstelliger Multiplex-Anzeige. Sollte eine entsprechende Anzeige bereits vorhanden sein, kann sie natürlich verwendet werden.

## Zusatzschaltung zur Programmierung von 2708 EPROM

Für die Programmierung von 2708-EPROM und 2716 der Firma TI (Mehrspannungstyp) werden die Adressen-, Daten- und Steuerleitungen auf eine zusätzliche Platine geführt. Zusätzlich ist noch eine −5-V- und +12-V-Versorgungsspannung und eine ca. +30-V-Programmierspannung notwendig.
Da es nicht ratsam ist, die 2708-EPROM mit nur einem Impuls zu programmieren, werden mit dieser Zusatzschaltung 90 Impulse à 1,5 ms erzeugt. Dies geschieht folgendermaßen:
Über IC22/23/24 wird eine Frequenz von 660 Hz erzeugt. Mittels IC25/29 kann das genaue Teilerverhältnis eingestellt werden. Daher wurde auf eine genaue Dimensionierung verzichtet. Es können Quarze bis max. 30 MHz verwendet werden, bei unter 2 MHz IC23 entfallen kann (Grenzfrequenz des 4020). Über einen Inverter des IC29 wird der Programmierimpuls des Grundgerätes zugeführt und schaltet das Flipflop IC26. Damit wird durch das IC26 die 660-Hz-Frequenz über T14/T15 in 26-V-Impulse umgewandelt. IC27/28 ist ein Teiler durch 90, welcher nach 90 Impulsen das Flipflop wieder rücksetzt. Danach ist der Programmiervorgang mit Schleifendurchlauf abgeschlossen.
Mit S3 kann wieder die Programmier/Lese-Auswahl erfolgen. Auch bei dieser Zusatzschaltung muß während des Einschaltens S3 in der „Lese"-Stellung sein!
Durch die LED D5 wird der Programmiervorgang optisch angezeigt. Die Adressen- und Datenleitungen werden direkt an den 2708 angeschlossen.

Hanspeter Povel

# EPROM-Programmer für Apple II/ITT 2020

## Einleitung

In der Grundausstattung des Apple II- bzw. ITT 2020-Mikrocomputers stehen keine parallelen Ein-/Ausgabeports zur Verfügung, wie dies bei vielen anderen ebenfalls auf dem Mikroprozessor 6502 basierenden Systemen, wie z.B. CBM, KIM, Atari, ATOM oder AIM der Fall ist. Für einfache Anwendungen kann zwar der sogenannte Game-Connector benutzt werden, der vier binäre Ausgänge (drei beim ITT 2020) und drei (zwei) binäre Eingänge besitzt. In komplizierteren Anwendungsfällen, wie z.B. der Ansteuerung von Druckern, Plottern oder Programmiergeräten werden jedoch meistens parallele 8-bit-Ein- und Ausgänge benötigt. Für diesen Zweck stehen aus der 6500-Mikroprozessorfamilie geeignete Interface-Bausteine, wie der 6520 PIA (Peripheral Interface Adapter) oder der 6522 VIA (Versatile Interface Adapter) zur Verfügung. Diese Schaltungen lassen sich ohne großen Hardwareaufwand in 6500-Systemen einsetzen. In der vorliegenden Bauanleitung wird gezeigt, wie ein 6522 VIA auch an einen Apple II- bzw. ITT 2020-Mikrocomputer angeschlossen werden kann. Um die universelle Verwendbarkeit dieser parallelen Ein-/Ausgabeschnittstelle zu demonstrieren, wird außerdem noch als Anwendungsbeispiel ein sehr einfaches Programmiergerät für 2716-EPROM beschrieben.

## Parallele Ein-/Ausgabe-Schnittstelle für den Apple II

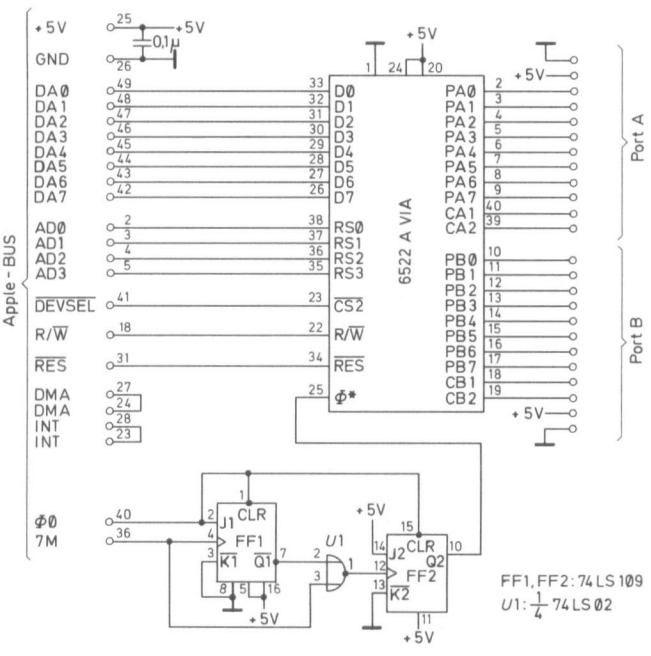
Der Anschluß an den Apple II-Bus erfolgt über einen der sieben 50poligen Gerätestecker (Slots). Jeder Slot läßt sich in einem bestimmten Adressenbereich durch das bereits dekodierte Signal $\overline{\text{DEVSEL}}$ (Pin 41) ansprechen, dabei wird dem N-ten Slot der Adreßbereich \$CØ8Ø + \$1Ø × (\$N + \$8) + \$X zugeordnet, wobei Ø ≦ \$X ≦ \$F und 1 ≦ \$N ≦ 7. Das Signal $\overline{\text{DEVSEL}}$ wird als Chip-Select CS für den 6522 verwendet. Da $\overline{\text{DEVSEL}}$ auf Null geht, während der 1-MHz-Systemtakt ΦØ hoch ist, kann ΦØ nicht wie sonst üblich als Takt für den 6522 benutzt werden. Aus diesem Grund wird mittels der beiden Flipflops FF1 und FF2 aus dem 7-MHz-Takt (7M, Pin 36) das Singal Φ* als Takt für den VIA erzeugt. Das Impulsdiagramm zeigt den zeitlichen Verlauf dieser Signale. Da Φ* nur 280 ns lang ist, muß für den VIA die schnellere Version 6522A verwendet werden.

Der Aufbau des Interfaces ist völlig unkritisch. Die beiden Ports A und B des VIA, sowie die dazugehörigen Steuersignale CA bzw. CB können getrennt auf je einen 16poligen IC-Sockel geführt werden. Die verbleibenden freien Leitungen lassen sich für Masse und Betriebsspannungen verwenden. Die Verbindung zur Außenwelt geschieht dann einfach durch Flachbandkabel mit passenden Steckern.

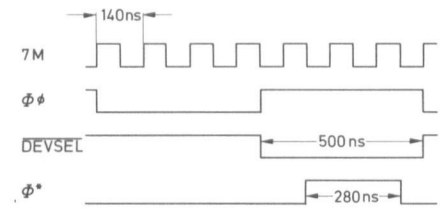
## Das EPROM-Programmiergerät

Die einfache Programmierbarkeit des inzwischen weitverbreiteten und preiswerten 2716-EPROM mit 2 K-Byte legt den Selbstbau eines Programmiergeräts nahe, das mit einem Mikrocomputer betrieben werden kann. Die gezeigte Schaltung, die zum Anschluß an das beschriebene Interface gedacht ist, zeichnet sich durch ihren äußerst geringen Aufwand an Hardware aus. Sie ist demnach einfach und preiswert aufzubauen.

Die EPROM-Adresse (AØ bis A1Ø) wird aus Port B des VIA abgeleitet. Die oberen vier Bit (A7 bis A1Ø) werden in dem Vier-

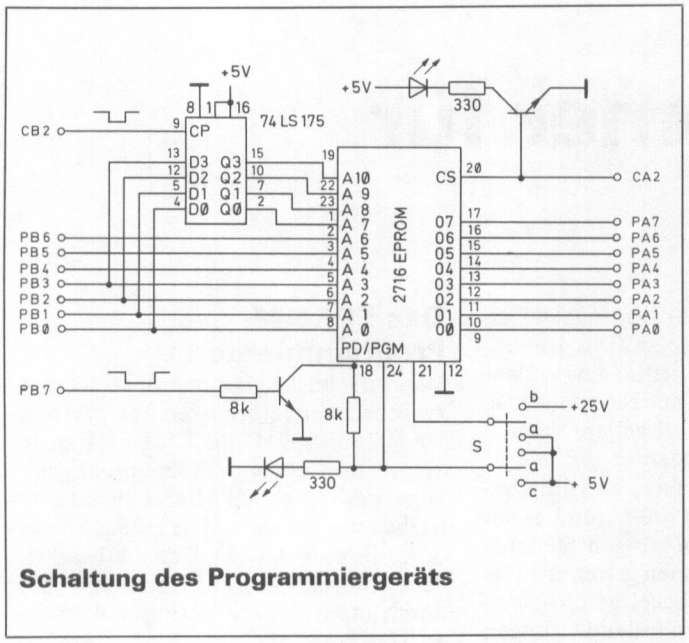



**Schaltung der 8-bit-parallelen Ein-/Ausgabeschnittstelle für den Apple-Mikrocomputer**

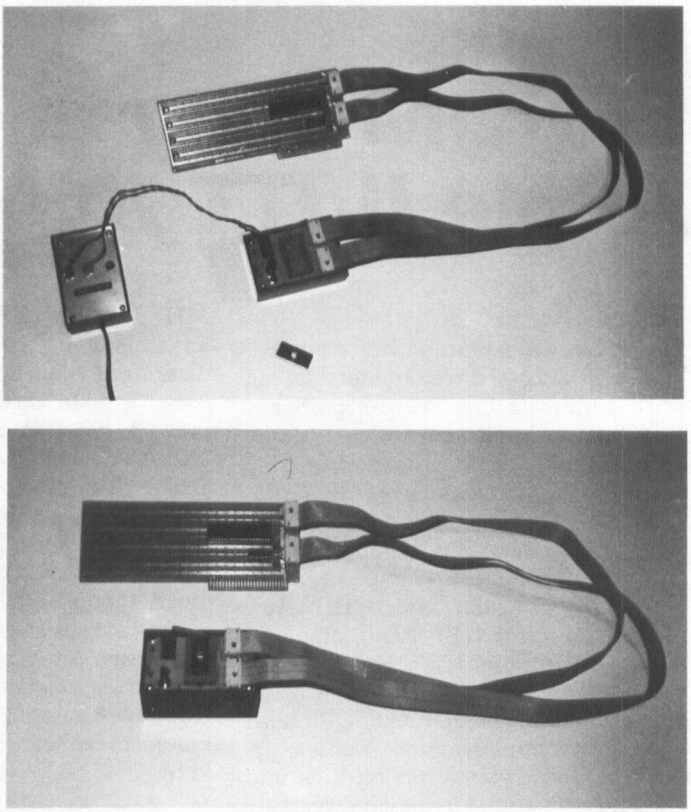



**Zeitlicher Verlauf der Taktsignale**
**7M: 7-MHz-Clock (Apple-Bus Pin 36)**
**ΦØ Mikroprozessor-Clock, Phase Ø**
**$\overline{\text{DEVSEL}}$: Device Select (Apple-Bus Pin 41)**
**Φ*: VIA-Clock**

# EPROM-Programmer für Apple II/ITT 2020




**Schaltung des Programmiergeräts**



**Aufbau des Interfaces und des EPROM-Programmiergerätes für den Apple II bzw. ITT 2020**

fach-D-Flipflop 74LS175 zwischengespeichert, während die unteren sieben Bit (AØ bis A6) direkt von PBØ bis PB6 kommen. CB2 wird als Taktsignal für die D-Flipflops verwendet. Der 50-ms-Programmierimpuls PGM wird vom Timer 1 des VIA erzeugt. Das Ausgangssignal von Timer 1 wird, wie weiter unten gezeigt, durch die Programmierung des VIA auf PB7 gelegt. Ebenso wird die Länge von PGM durch die Programmierung des VIA bestimmt. Das Chip-Select-Signal $\overline{CS}$ wird von CA2 gebildet. Die Daten des EPROM werden über Port A des VIA in beiden Richtungen gesteuert. Der Schalter S dient zum Abschalten der Betriebsspannung (Stellung a) und zum Umschalten zwischen dem Programmierzustand (Stellung b) und dem Lesezustand (Stellung c) des EPROM.
Die Programmierung des Geräts kann einfach durch ein BASIC-Programm erfolgen, dessen Gestaltung weitgehend frei ist. Es seien hier nur einige Hinweise zur Programmierung des VIA gegeben.

Für das EPROM-Programmiergerät müssen verschiedene Register des VIA geladen werden. Die Adressen dieser Register werden folgendermaßen aus der Slot-Nummer berechnet:

| | |
|---|---|
| NS: | Slot-Nummer (zwischen 1 und 7) |
| DEV = 49152 + 16 (8 + NS): | Device-Basisadresse |
| OB = DEV: | Datenregister A |
| OA = DEV + 1: | Datenregister B |
| DB = DEV + 2: | Datenrichtungsregister B |
| DA = DEV + 3: | Datenrichtungsregister A |
| L1 = DEV + 4: | unteres Byte von Timer 1 |
| H1 = DEV + 5: | oberes Byte von Timer 1 |
| AC = DEV + 11: | Hilfsregister |
| PC = DEV + 12: | Peripheres Kontrollregister |
| UF = DEV + 13: | Unterbrechungs-Flagregister |

Zum Initialisieren des Interfaces, sowie zum Lesen, bzw. Programmieren des EPROM sind die Register des VIA folgendermaßen zu laden (siehe Datenblatt des 6522 VIA):

a. Initialisierung

| | |
|---|---|
| POKE DB, 127: | PBØ bis PB6 sind Ausgänge |
| POKE AC, 128: | Timer 1 im One-Shot-Mode an PB7 |

b. Lesen des EPROM

| | |
|---|---|
| POKE DA,Ø: | PAØ bis PA7 sind Eingänge |
| POKE PC, 172: | $\overline{CS}$ = L, CB2 im Pulsmode |
| POKE OB, AH: | Lade D-Flipflops mit den oberen 4 Adreßbits |
| POKE PC, 236: | $\overline{CS}$ = L, CB2 = H |
| POKE OB, AL: | Lade die unteren 7 Adreßbits |
| DAT = PEEK (OA): | Lese die Daten unter der zuvor eingegebenen Adresse (AL, AH) |

c. Programmieren des EPROM

| | |
|---|---|
| POKE DA, 255: | PAØ bis PA7 sind Ausgänge |
| POKE PC, 174: | $\overline{CS}$ = H, CB2 im Pulsmode |
| POKE OB, AH: | Lade D-Flipflops mit den oberen 4 Adreßbits |
| POKE PC, 238: | $\overline{CS}$ = H, CB2 = H |
| POKE OB, AL: | Lade die unteren 7 Adreßbits |
| POKE OA, DAT: | Programmiere die Daten an der zuvor eingegebenen Adresse (AL, AH) |
| POKE H1, 195: | Starte Timer 1, 50 ms Puls |
| PEEK (UF) < 64: | während des Programmierpulses |
| PEEK (UF) = 64: | am Ende des Programmierpulses |

```
]LIST

1  DIM SW$(8),BF%(2047),G(4),C$(15)
2* DATA     "D","V","P","C","L","T","E","H"
3  FOR L = 1 TO 8: READ SW$(L): NEXT
4 G(1) = 4096:G(2) = 256:G(3) = 16:G(4) = 1
5  DATA  "0","1","2","3","4","5","6","7","8",
       "9","A","B","C","D","E","F"
6  FOR L = 0 TO 15: READ C$(L): NEXT
10  HOME
20  PRINT "2716 EPROM PROGRAMMER"
25  PRINT "========================"
30  INPUT "SLOT NUMBER? ";NS
40 DEV = 49152 + 16 * (8 + NS)
50 OB = DEV:OA = DEV + 1
60 DB = DEV + 2:DA = DEV + 3
70 L1 = DEV + 4:H1 = DEV + 5
80 AC = DEV + 11:PC = DEV + 12
90 UF = DEV + 13
100  POKE DB,127: POKE AC,128
105  REM                              - * -
110  PRINT : PRINT
120  PRINT "OPTIONS:"
130  PRINT "D: DATA INPUT"
140  PRINT "V: VERIFY PROGRAM"
150  PRINT "P: PROGRAM"
160  PRINT "C: COPY EPROM INTO BUFFER"
170  PRINT "L: LIST BUFFER"
180  PRINT "T: TEST IF EPROM ERASED"
190  PRINT "E: EXIT"
200  PRINT "H: HELP"
210  PRINT "========================"
220  PRINT : PRINT
300  INPUT "* MODE? ";MS$
310  FOR L = 1 TO 8
320  IF MS$ = SW$(L) GOTO 350
330  NEXT
340  GOTO 300
350  IF MS$ = "E" GOTO 9999
360  IF MS$ = "T" GOTO 490
370  IF MS$ = "H" GOTO 110
400  PRINT "  1. ADDR. ";
405  GOSUB 9000:A1% = Z
410  IF MS$ = "D" GOTO 1000
415  PRINT "  2. ADDR. ";
420  GOSUB 9000:A2% = Z
430  IF MS$ = "L" GOTO 2000
440 A% = A1%
450 AH% =  INT (A% / 128)
460 AL% = A% - AH% * 128
470  IF MS$ = "P" GOTO 3000
480  GOTO 500
490 A1% = AL% = AH% = 0:A2% = 2047
500  POKE DA,0
510  POKE PC,172: POKE OB,AH%
520  POKE PC,236
530  POKE OB,AL%
540 B% =  PEEK (OA)
550  IF MS$ <  > "T" GOTO 590
560  IF B% = 255 GOTO 630
570  PRINT "  ERROR AT ADDR. ";
575 L0 = 1:Z = A%: GOSUB 9100: PRINT : PRINT
580  GOTO 300
590  IF MS$ <  > "V" GOTO 620
600  IF B% <  > BF%(A%) GOTO 570
610  GOTO 630
620 BF%(A%) = B%
630 A% = A% + 1
640  IF A% > A2% GOTO 690
650 AL% = AL% + 1
660  IF AL% < 127 GOTO 530
670 AL% = 0:AH% = AH% + 1
680  GOTO 510
690  PRINT "  DONE": PRINT : CALL 64477
700  GOTO 300
800  REM                              - * -
1000 A% = A1%
1010  PRINT "  ";:L0 = 1:Z = A%
1015  GOSUB 9100
1020  INPUT ": ";XI$
1030  IF XI$ = "/" GOTO 690
1035  GOSUB 9020:BF%(A%) = Z
1040 A% = A% + 1
1050  IF A% > 2047 GOTO 690
1060  GOTO 1010
1070  REM                              - * -
2000 A% = A1%
2010  PRINT
2015 L0 = 1:Z = A%: GOSUB 9100
2020  PRINT ": ";
2030 K = 0
2040 L0 = 3:Z =
     BF%(A%): GOSUB 9100: PRINT " ";
2050 A% = A% + 1
2060  IF A% > A2% THEN
      PRINT : PRINT : GOTO 300
2070 K = K + 1
2080  IF K = 8 GOTO 2010
2090  GOTO 2040
2095  REM                              - * -
3000  POKE DA,255
3010  POKE PC,174: POKE OB,AH%
3020  POKE PC,238
3030  POKE OB,AL%
3040  POKE OA,BF%(A%)
3050  POKE L1,0: POKE H1,195
3060 A% = A% + 1
3070  IF A% > A2% THEN  POKE DA,0:
      POKE PC,236: GOTO 690
3080 AL% = AL% + 1
3085  IF  PEEK (UF) < 64 GOTO 3085
3090  IF AL% < 127 GOTO 3030
3100 AL% = 0:AH% = AH% + 1
3110  GOTO 3010
9000  REM                              - * -
9005  REM   CONVERT HEX INPUT INTO DEC
9010  INPUT "? ";XI$
9020 Z = 0:L =  LEN (XI$)
9030  IF L > 4 GOTO 9010
9040  FOR K = 1 TO L
9050 N =  ASC ( MID$ (XI$,K,K)) - 48
9060  IF N > 9 THEN N = N - 7
9070  IF N > 15 GOTO 9010
9080 Z = Z + N * 16 ^ (L - K): NEXT
9090  IF Z > 2047 GOTO 9010
9095  RETURN
9100  REM                              - * -
9105  REM   CONVERT DEC OUTPUT INTO HEX
9110  FOR L = L0 TO 4
9120 NZ =  INT (Z / G(L)): PRINT C$(NZ);
9130 Z = Z - NZ * G(L): NEXT
9140  RETURN
9999  END
```

Axel Schulze

# Systemerweiterung für KIM1

Das KIM1-Mikroprozessorsystem besitzt in seiner Grundversion einen dekodierten Speicherblock von 8 KByte ($\overline{K0}$ bis $\overline{K7}$), wobei die mit $\overline{K1}$ bis $\overline{K4}$ bezeichneten 4 KByte nicht bestückt sind.

Meine Systemerweiterungskarte entstand aus dem Gedanken, zum einen diese 4 KByte mit zwei EPROM-Bausteinen des Typs 2716 und eigenem Monitorprogramm zu bestücken. Zum anderen weitere, im Handel erhältliche Platinen mit dem VG64-Stecker an den KIM anschließen zu können.

Hierbei ergibt sich die dritte sehr nützliche Erweiterung. Die Möglichkeit nämlich, Interrupt-Vektoren von der CPU nicht nur aus dem mit $\overline{K7}$ dekodierten ROM des MPS 6530-002 zu holen, sondern mit einem DIP-Schalter wählbar, aus der zero-page ($\overline{K0}$) oder einem der beiden EPROM ($\overline{K2}$ oder $\overline{K4}$). So muß der Prozessor nach einem Reset nicht mehr unbedingt das KIM-Monitorprogramm abarbeiten, sondern startet sofort mit einem Programm des Anwenders, dessen Startadresse von den oberen Adressen eines der beiden EPROM geholt wird.

Die Schaltung kann auf einer Lochrasterplatine im Euroformat aufgebaut werden, an deren Vorderseite sich der Stekker zum „application connector" des KIM befindet, ihm gegenüber ist ein VG64-Stecker eingelötet, an den eine Busplatine angeschlossen wird.

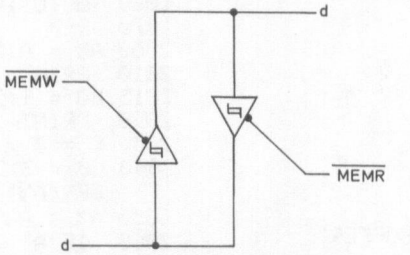
## Die Adressen- und Datenbuspufferung

Zur Pufferung des Adressen- und Datenbusses dienen je zwei nichtinvertierende Tri-State-Buffer 74LS244 J6 bis J9. Die Steuerleitungen $\overline{1g}$ und $\overline{2g}$ liegen bei J8 und J9 an Masse, somit puffert jeder Baustein acht Leitungen des Adressenbusses unidirektional. Je vier Leitungen des Datenbusses liegen an J6 und J7. Für den bidirektionalen Datenverkehr ist der Eingang eines Verstärkers mit dem Ausgang des anderen verbunden. Durch die Signale $\overline{MEM\ W}$ an der Steuerleitung $\overline{1g}$ und $\overline{MEM\ R}$ an $\overline{2g}$ schaltet der Puffer immer dann zwischen der Richtung Lesen und der Richtung Schreiben um, wenn die mit KØ, K5, K6 und K7 dekodierten Speicher der KIM-Platine nicht angesprochen werden.

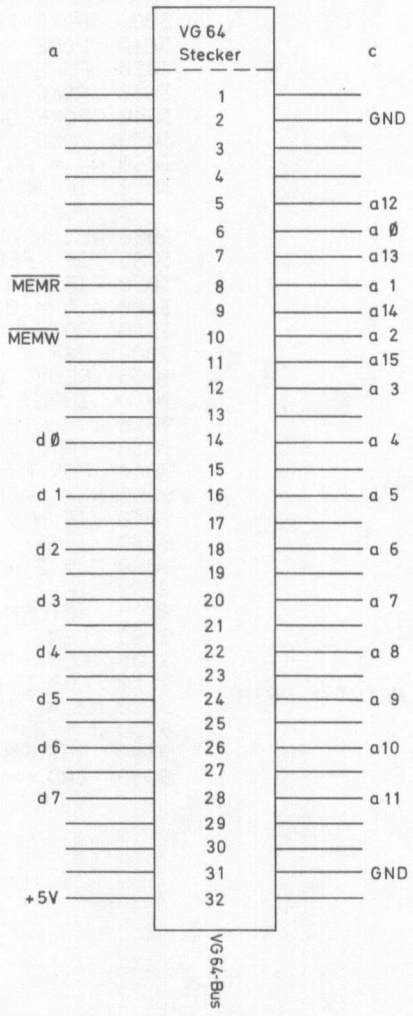
## Die Erzeugung der $\overline{MEM\ R}$-, $\overline{MEM\ W}$-Signale

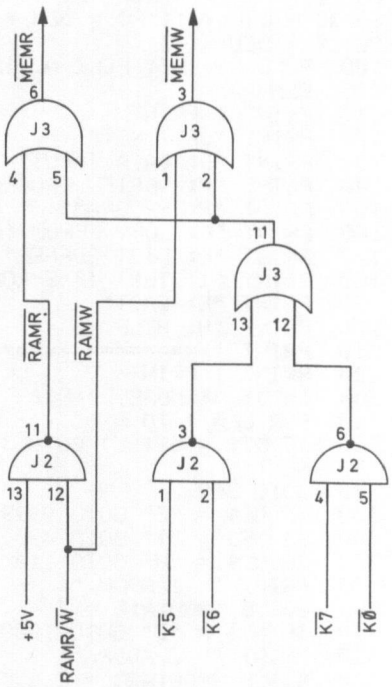
Zu diesem Zweck liefert der KIM ein RAM-R/W-Signal, das über J2,12 (J2 = Nummer des IC im Schaltplan, 12 = Pin dieses IC) in ein $\overline{RAM\ R}$- und ein $\overline{RAM\ W}$-Signal aufgespalten wird. Das NAND-Gatter J2 arbeitet hier als Inverter, da sein zweiter Eingang J2,13 an +5 V liegt.

Am ODER-Gatter J3,6 (bzw. J3,3) ent-

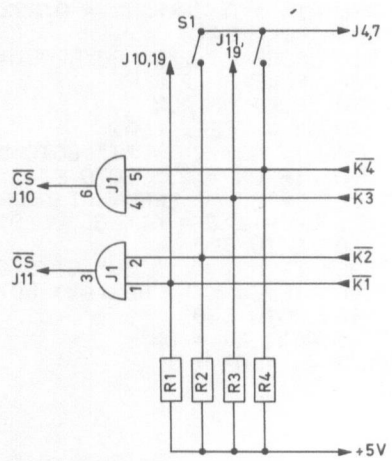



**Schaltung des Buffers 74LS244 für den bidirektionalen Datenfluß**

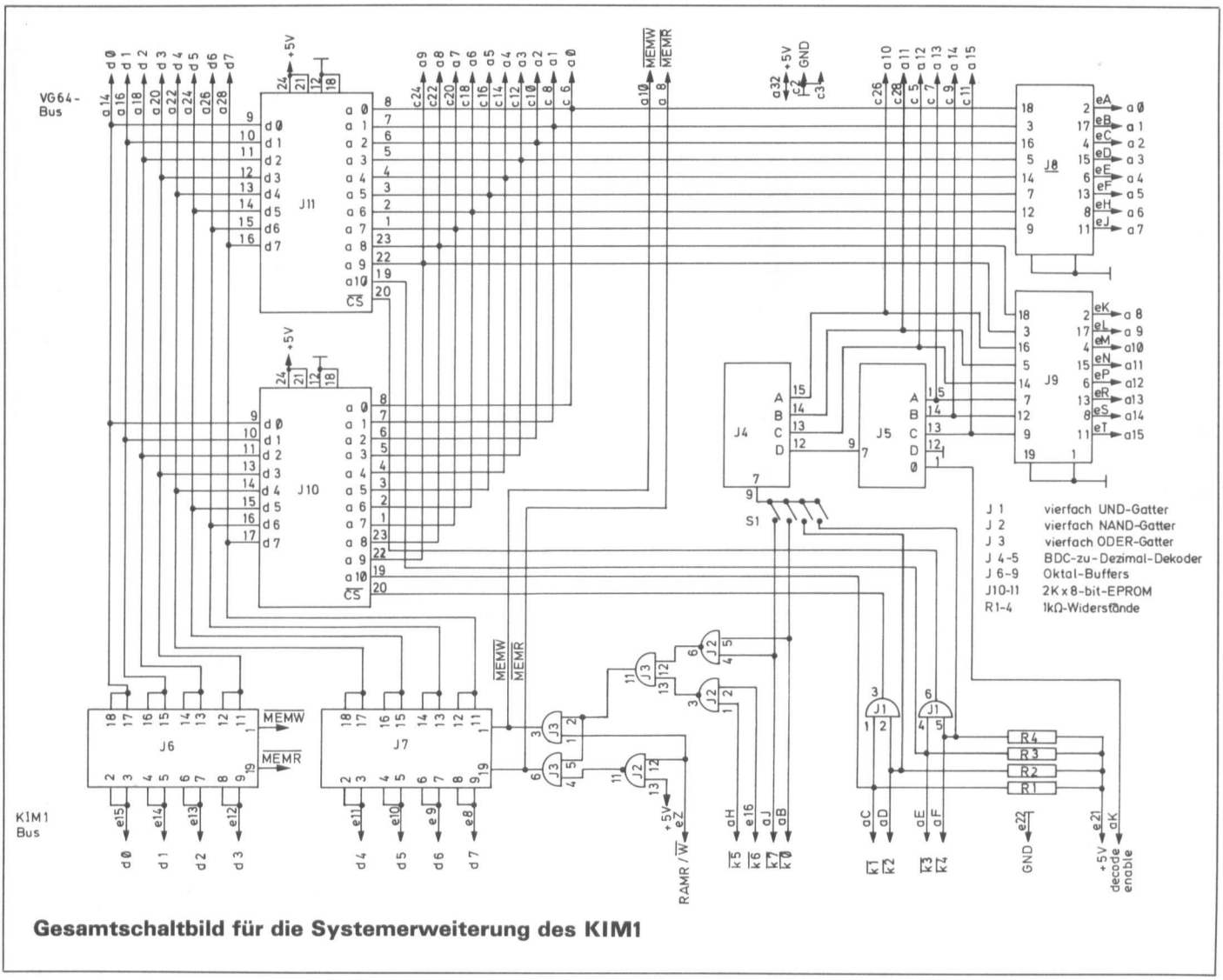



**Erzeugung von den Steuersignalen $\overline{MEW\ W}$, MEM R**




**Verarbeitung der Dekoder-Signale $\overline{K1}$ bis $\overline{K4}$**

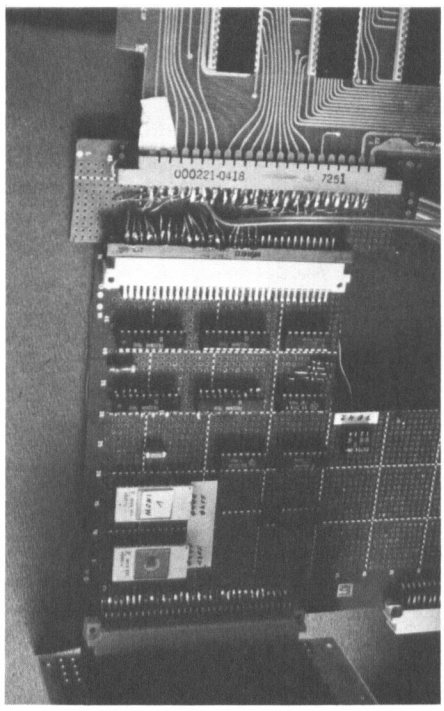



**Gesamtschaltbild für die Systemerweiterung des KIM1**

steht das $\overline{\text{MEM R}}$- (bzw. $\overline{\text{MEM W}}$-) Signal durch Verknüpfung des RAM R- (bzw. $\overline{\text{RAM W}}$-) Signals mit dem Ausgang eines vierfach NAND-Gatters, an dessen Eingängen die vier Dekodersignale KØ, $\overline{\text{K5}}$, $\overline{\text{K6}}$ und $\overline{\text{K7}}$ liegen.
Anstelle des vierfach NAND arbeitet meine Karte mit einer Kombination aus 2/4 NAND J2 und $^1/_4$ ODER-Gatter J3 mit der gleichen Funktion: Die Datenbuspuffer sind dann in beiden Richtungen gesperrt, wenn der Prozessor auf Speicher der KIM-Platine zugreift.

## Anschluß der zwei 2K × 8-bit-EPROM 2716

Die zwei EPROM belegen den Speicherbereich (hex) Ø4ØØ – Øbff (J10) und ØcØØ – 13ff (J11). Beide liegen parallel am 8-bit-Datenbus dØ – d7 und am 10-bit-Adressenbus aØ – a9. Zur Erzeugung der $\overline{\text{CS}}$-Signale (chip-select) werden die über je einen Widerstand $R_1 - R_4$ auf +5 V



**EPROM- und Systemerweiterungskarte als Bindeglied zwischen VG64-Bus und KIM1**

hochgezogenen Dekoder-Signale $\overline{\text{K1}}$ und $\overline{\text{K2}}$ für J10,20 sowie $\overline{\text{K3}}$ und $\overline{\text{K4}}$ für J11,20 über zwei UND-Gatter J1 zusammengeschaltet.
Die von den IC geforderte Adressenleitung a1Ø liegt bei J10,19 an $\overline{\text{K1}}$, bei J11,19 an $\overline{\text{K3}}$. Besitzt man EPROM, die mit invertiertem a1Ø programmiert sind, genügt es, $\overline{\text{K1}}$ mit $\overline{\text{K2}}$ und $\overline{\text{K3}}$ mit $\overline{\text{K4}}$ zu vertauschen.
Als Spannungen liegen an Pin 24 ($V_{cc}$) und 21 ($V_{pp}$) + 5 Volt, an Pin 18 ($\overline{\text{CE}}$, chip enable) und 12 (GND) Ø Volt.

## Verbindung zum VG64-Bus

Wie schon erwähnt, befindet sich am Ende meiner Systemerweiterungskarte ein VG64-Stecker, auf den eine Bus-Platine aufzustecken ist.

# CHIP-BÖRSE

Unter dieser Rubrik veröffentlicht CHIP Textanzeigen zu einem besonders günstigen Preis.

Nutzen Sie den Anzeigen-Service, wenn Sie

- Hardware oder Software verkaufen wollen
- Programme austauschen möchten
- Kontakte und Erfahrungsaustausch anstreben
- Literatur suchen oder anbieten

Private Gelegenheitsanzeigen je Druckzeile 6 DM inkl. MwSt.

Musteranzeige

| |
|---|
| Suche Mini-Printer, Ansteuerelektronik, möglichst 64 Zeichen/Bit parallel Eingang/Zeichen seriell. H. J. Kraft, S 17/68 Mannheim |

nur 24 DM

Gewerbliche Gelegenheitsanzeigen je Druckzeile 8 DM zuzügl. MwSt.

Musteranzeige

| |
|---|
| Verkaufe Datensichtgeräte 80 × 24 Z, VB 750,– DM. Mikrocomputer-Kits, Peripherie-Software äußerst günstig. H. Jung, Telefon (0 40) 31 46 78 |

nur 32 DM zuzügl. MwSt.

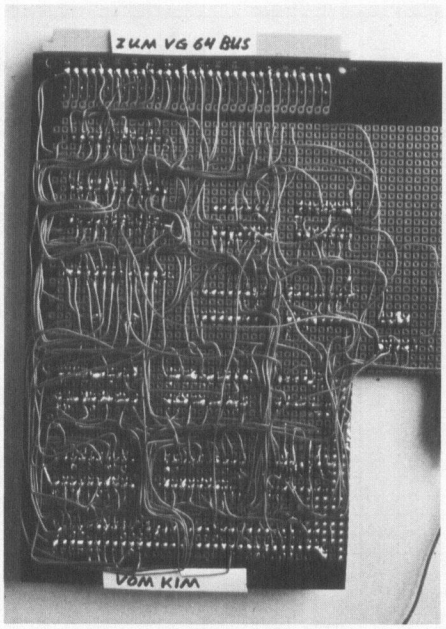
Chiffregebühr 6 DM inkl. MwSt.

CHIP

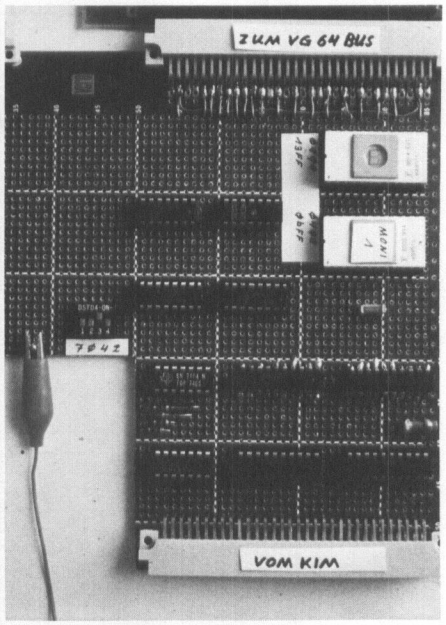
Anzeigenservice
Vogel-Verlag
Postfach 67 40

D-8700 Würzburg 1

# Systemerweiterung für KIM1




**Unterseite, mit Verdrahtung**




**Bauteileseite der Platine**

## Dekodierung und Interrupt-Vector-Select

In der Grundversion des KIM1-Systems werden zur Dekodierung des unteren 8K-Speicherblocks nicht die Adressenleitungen a13 bis a15 berücksichtigt. Der Eingang d (decode enable) des platineninternen Dekoders 74LS145 liegt an Masse. Somit wiederholt sich dieser Speicherblock siebenmal in den 64 KByte, die der Prozessor insgesamt ansprechen kann.

Die erweiterte Dekodierung meiner Karte hat die Aufgabe, eine Speichererweiterung über die vorhandenen 8 KByte hinaus zu ermöglichen und einen frei wählbaren Speicherbereich in das 64 KByte zu kopieren, um so die Interrupt- und Reset-Verarbeitung sicherzustellen.

Hierzu benötigt man zwei weitere Dekoder des Typs 74145 (J4 und J5). J5, dessen Eingänge an a13 bis a15 liegen, dekodiert die oberen und die unteren 8 KByte des Systems, sein Ausgang Ø (J5,1) liegt am Eingang d des KIM-Dekoders. Somit wird eine Wiederholung dieses Bereichs nicht mehr zugelassen, allerdings ist jetzt auch kein Reset mehr möglich.

Der Ausgang J5,9 liegt am Eingang d des dritten Dekoders J4, dessen Ausgang J4,9 das oberste 1 KByte $\overline{K63}$ dekodiert.

Schaltet man nun mit einem DIP-Schalter diesen Ausgang auf die vom KIM kommenden Dekoder-Signale (KØ, $\overline{K2}$, $\overline{K4}$ oder $\overline{K7}$ wird dieses 1 KByte in $\overline{K63}$ kopiert, der Prozessor findet die Interrupt-Adressen und beginnt, ist $\overline{K7}$ eingeschaltet, mit dem KIM-Monitorprogramm. Bei eingeschaltetem $\overline{K2}$ oder $\overline{K4}$ startet beim Reset ein Anwenderprogramm, dessen Startadresse aus dem oberen Speicherbereich des betreffenden EPROM geholt wird. Zum Testen von verschiedenen Resets vor dem Einbrennen in ein EPROM wird der mit $\overline{K\emptyset}$ bezeichnete RAM-Bereich des KIM verwendet.

## Wahrheitstabellen

Chip-select-Signale:

| J1,1 $\overline{K1}$ | J1,2 $\overline{K2}$ | J1,4 $\overline{K3}$ | J1,5 $\overline{K4}$ | J1,3 $\overline{CS}$,J10 | J1,6 $\overline{CS}$,J11 |
|---|---|---|---|---|---|
| L | H | H | H | L | H |
| H | L | H | H | L | H |
| H | H | L | H | H | L |
| H | H | H | L | H | L |
| H | H | H | H | H | H |

Verknüpfung von $\overline{K\emptyset}$, $\overline{K5}$, $\overline{K6}$, $\overline{K7}$:

| J2,1 | J2,2 | J2,4 | J2,5 | J3,11 |
|---|---|---|---|---|
| L | H | H | H | H |
| H | L | H | H | H |
| H | H | L | H | H |
| H | H | H | L | H |
| H | H | H | H | L |

Erzeugung von $\overline{MEM\ W}$

| J3,1 $\overline{RAM\ W}$ | J3,2 von J3,11 | J3,3 $\overline{MEM\ W}$ |
|---|---|---|
| H | H | H |
| H | L | H |
| L | H | H |
| L | L | L |

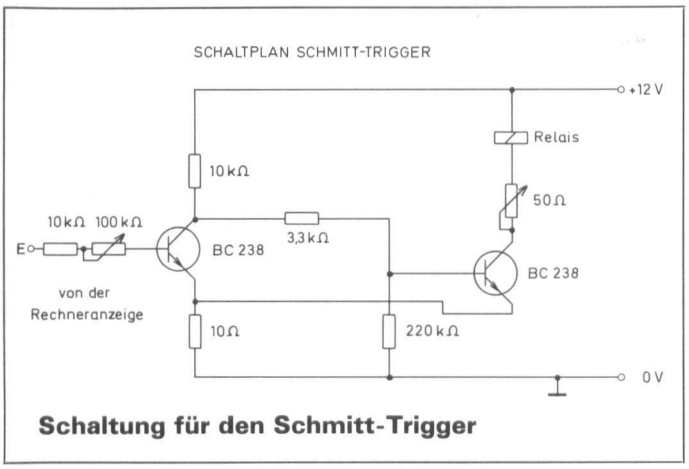
Identisch mit $\overline{MEM\ R}$-Erzeugung.

Matthias Poppe und Holger Dierks

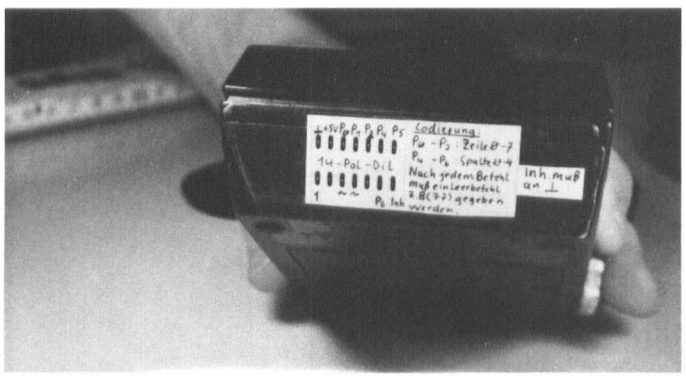
# KIM1 steuert SR51

Die Problemstellung ist und war, einen nichtprogrammierbaren Taschenrechner (Texas Instruments SR51) mit Hilfe eines Mikrocomputers (KIM1) zu steuern. Damit wird der SR51 in einem gewissen Sinne programmierbar. Wir überlegten uns, die Tastendrücke durch den Computer zu steuern. Als Lösung boten sich Demultiplexer (4051) an, die über eine Treiberstufe (7407) an die Computerports angeschlossen wurden. Der eine Demultiplexer ist zur Ansteuerung der Zeilen, der andere zur Ansteuerung der Spalten nötig. Aufgebaut haben wir das Ganze auf eine Lochrasterplatine und die Schaltung noch im Taschenrechner selbst untergebracht.

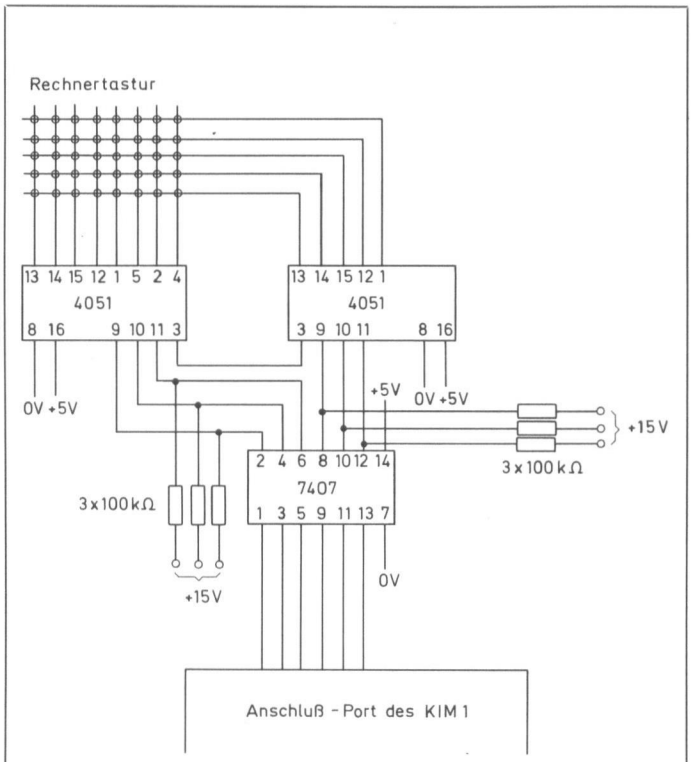
Nun kann der Computer schon jede Taste des Rechners durch ein einfaches Programm (s. Basisprogramm 5) „drücken".



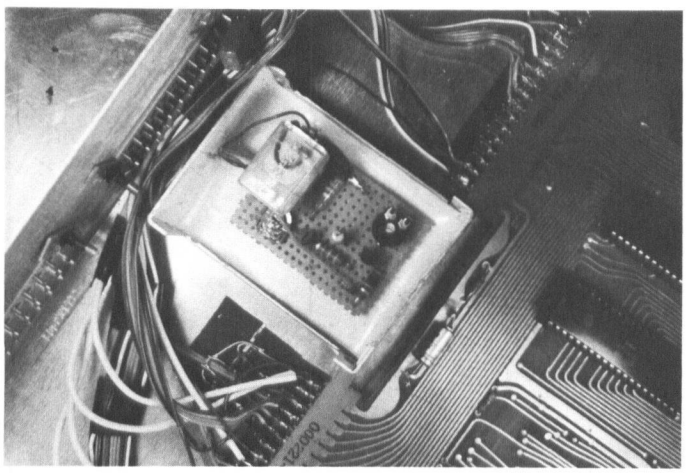

**Schaltung für den Schmitt-Trigger**



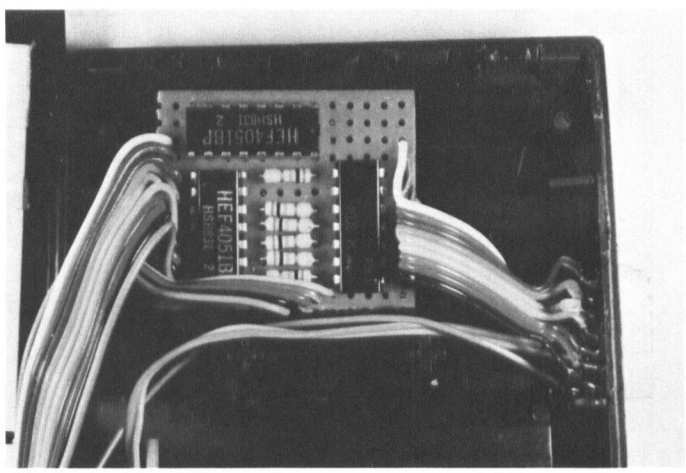
**Pinbelegung für den Taschenrechner SR51**



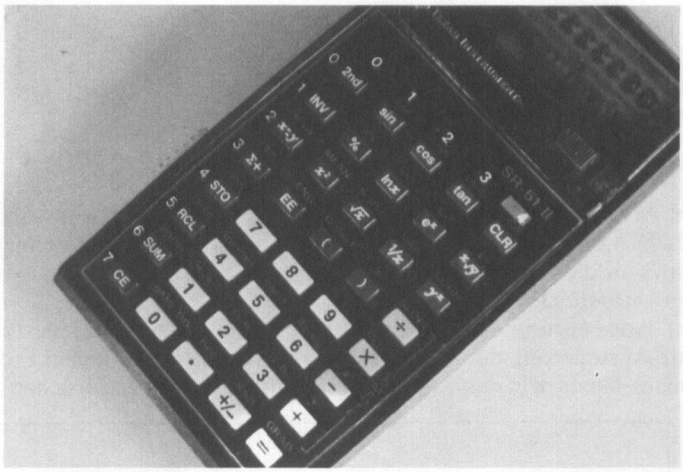

**Interface zwischen KIM1 und SR 51 mit den elektronischen Schaltkreisen**



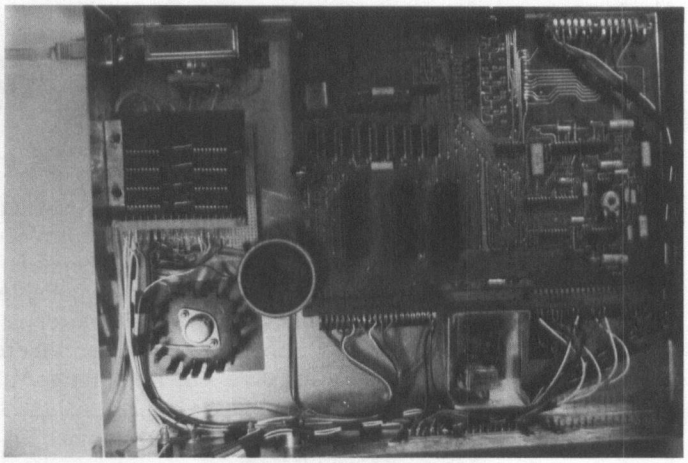
**Interface auf der KIM1-Platine montiert**




**Gesamtansicht der Interface-Platine**




**Taschenrechner SR51 II**



**Platine des Computers KIM1**

Dazu muß man nur die Ports mit der entsprechenden Tastennummer laden.

## Die Realisierung

Damit der Computer aber auch entscheiden kann, ob ein Wert >, < oder =0 ist, verwendeten wir einen kleinen Trick: Steht die Zahl in der Anzeige, läßt man die Taste ln x drücken, und für $x \leq 0$ blinkt die Anzeige, für $x > 0$ bleibt sie „stehen". Drückt man die Taste 1/x, blinkt die Anzeige für $x = 0$.

Es ist also das Blinken der Anzeige abzufragen, welches durch einen Schmitt-Trigger geschieht, der aus Platzgründen im selbstgebauten Computergehäuse untergebracht ist. Er ist ebenfalls einfach auf einer Lochrasterplatine aufgebaut.

Uwe Henning

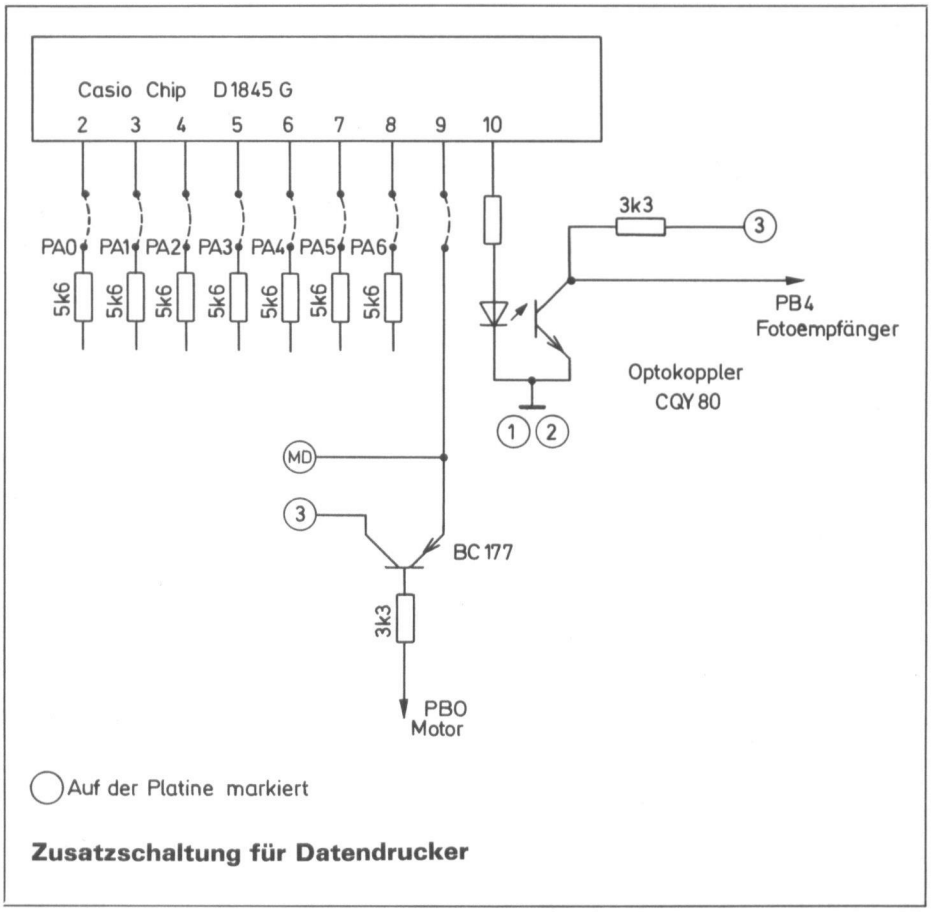
# Preiswerter Datendrucker

Als ich mir den CASIO HR-10 kaufte, brauchte mir das Fräulein an der Kasse keinen Garantieschein auszuschreiben, denn ich wußte, daß wird mein Datendrucker.
Nach einer Stunde war die Hardware fertig. Dieser Elektronikrechner mit Drukker arbeitet in Ätzbeizdruck und hat sieben Nadeln. Die Mechanik dieses Rechners ist sehr robust und einfach aufgebaut.
Auch die Lösung der Schnittstelle meines Mikrocomputers und des HR-10 ist mit Hilfe eines Draht-PROM zu jeder Zeit für Taschenrechnerbetrieb oder Datendrucker vorgesehen.

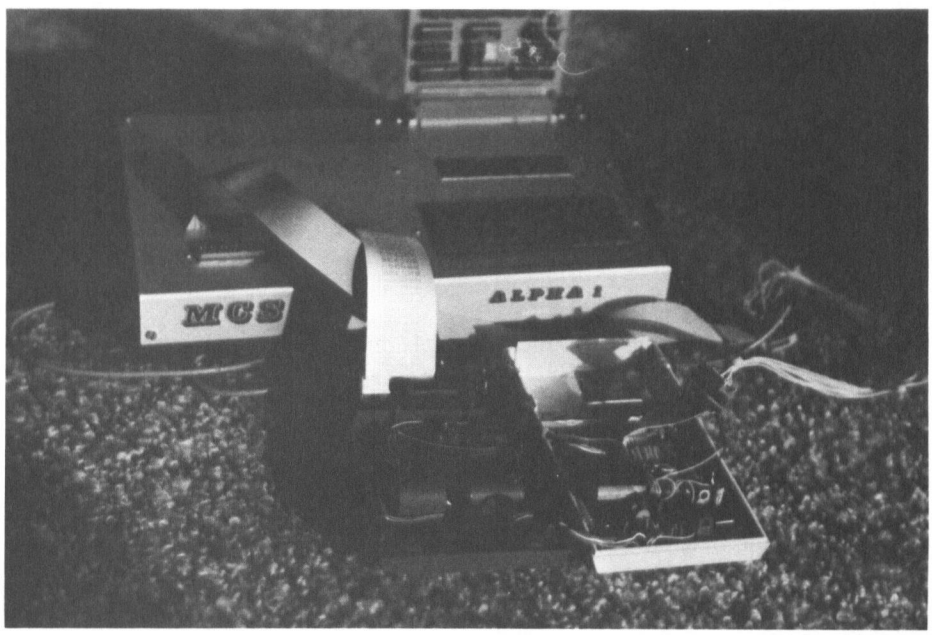
## Die Schnittstellenbeschreibung

Für die Ansteuerung der Nadeln wird die gesamte Elektronik des Taschenrechners

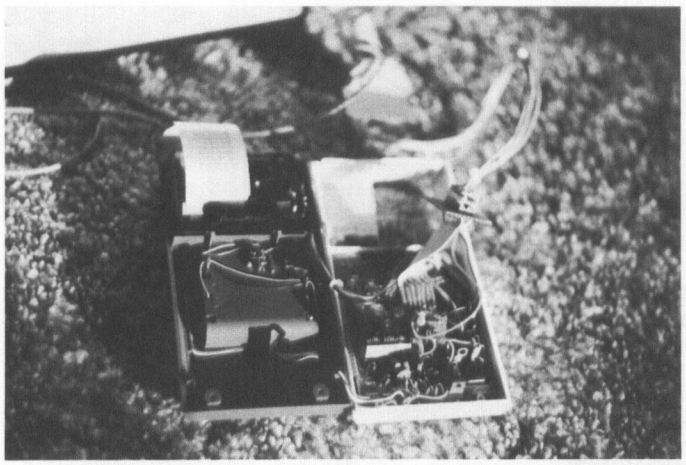
```
A9FF8D01F6A9
0F8D03F6A9F0
8D02F6203502
20402203502
A9E0CD02F6D0
10204802203S
02A9F0CD02F6
D0034C1202A9
F18D02F660A9
558501C601D0
FC60EAEAA9E0
CD02F6D0F960
A9F0CD02F6D0
F960A9F08D02
F6204802A9B7
8508204802C6
00F006204802
4C5C02A9F18D
02F660602000
02A205A9F08D
02F6204802BD
16008502200O
F7CAF0062048
024CT5026019
A9078540A202
A007840D89F8
01851088EAEA
EAEAEAEAEAEA
EAEAEAEAEAEA
EAEACA8650F0
03966360A207
18A540690685
40A8865O4C93
020000000600
000000000600
```



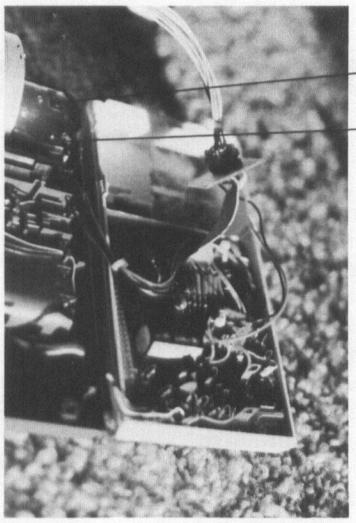

**Zusatzschaltung für Datendrucker**

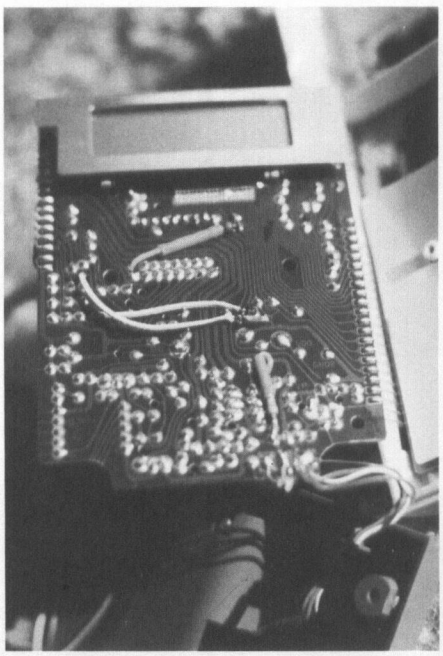



**Normalerweise kosten Datendrucker über 200 DM. Durch die Verwendung eines Elektronikrechners mit Druckwerk läßt sich der Preis auf etwa 120 DM realisieren. Das Bild zeigt den CASIO HR-10 mit dem MCS-System ALPHA 1**



**Die herausgelöteten 5,6-kΩ-Widerstände (mit rotem Schlauch geschützt) werden durch ein Flachbandkabel zum IC-Sockel geführt. Zu erkennen ist auch die Miniplatine mit dem Optokoppler**



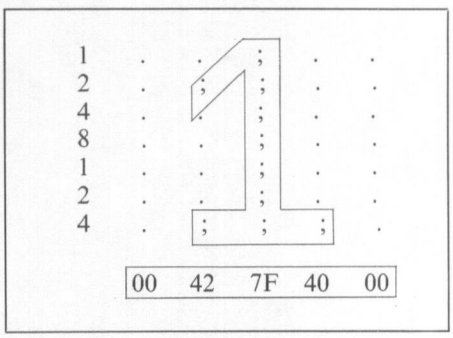



**Lötseite der HR 10-Platine. Die beiden Widerstände 100 Ω sind oben zu erkennen und 3,3 kΩ unten rechts**

**Drucker komplett mit Interface, wobei links die Verbindungsleitungen sind**

## Die Programmbeschreibung

- Initialisierung; PORT definieren
- Nullpunkt suchen
- Zahl suchen
- Da der Drucker von rechts nach links arbeitet, muß man (bei 6 Stellen) mit der 6. Stelle anfangen, bis 0 und dann bei der 12. Stelle bis 6 usw. Zahl verschieben
- Druckkode aus Tabelle holen
- Drucken
- Beispiel des Druckkodes für 1




mitverwendet. Man braucht nur die sieben Widerstände, die nach dem Rechnerchip, dem Pin 2 – 8, angeordnet sind, auf der Chipseite auszulöten und an die PIA des Mikrocomputers anzulegen. Ich habe mich für PA0 bis PA6 entschieden.
Der HR-10 besitzt eine Lochscheibe, die mit dem Motorvorschub gekoppelt ist. Diese Lochscheibe wird mit einer Lichtschranke abgefragt. So ist eine Information der jeweiligen Stellung des „Nadelschlittens" möglich. Hat der Schlitten die Endstellung erreicht, wird ein Hebel betätigt, der die Lichtschranke verdunkelt. Diese Hell-Dunkel-Information greife ich mir mit einem Optokoppler ab und gebe sie an den PB4 meines Computers. Für die Motoransteuerung wird zum Schutz der PIA PBO ein BC177 vorgeschaltet.
Das Programm ist speziell für den Alpha 1 geschrieben. Dieser Mikrocomputer arbeitet mit dem 6502.
Durch geringfügige Änderungen läuft dieses Druckprogramm auch auf andere 6502-Systeme. Man braucht nur die PIA-Adressen zu ändern. Dieses Programm, welches ich ausgedruckt habe, ist natürlich schön hintereinander in einem EPROM auf Adresse E000 gespeichert.

Für diesen preiswerten Datendrucker wurde ein Gerät von CASIO verwendet. Selbstverständlich lassen sich auch Drucker von anderen Herstellern verwenden.
Dieser Selbstbau bringt dem Hobby-Anwender nicht nur einen preisgünstigen Drucker, sondern auch einen Taschenrechner, wenn mit Optokopplern gearbeitet wird.
Bei der Verwendung von anderen Druckern muß die Zusatzschaltung etwas geändert werden.

Klaus Musseleck

# Analogdaten-Aufzeichnungssystem für Langzeitmessung

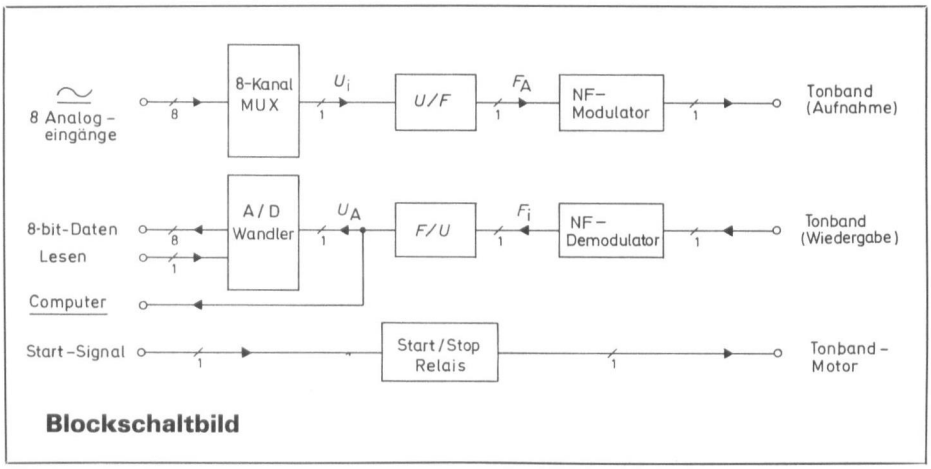
In zahlreichen Gebieten der Meß- und Regelungstechnik kommt es vor, daß vor Ort mehrere analoge Meßsignale über einen längeren Zeitraum beobachtet und später nach bestimmten Verfahren durch einen stationären Rechner ausgewertet werden sollen.
Bei derartigen zum Teil sich nur langsam ändernden Meßgrößen können die interessanten Zeitintervalle im Minuten- oder Stundenbereich liegen, so daß ein stichprobenartiges Registrieren in bestimmten Zeitzyklen ausreicht.
Die vorliegende Arbeit beschreibt ein System, mit dem bis zu acht langsame Analogkanäle gleichzeitig registriert und auf einen Kassettenrecorder aufgezeichnet werden. Die Daten werden in ein gemultiplextes Analogsignal zurückgewandelt, das über einen A/D-Wandler durch einen Mikrocomputer mit 8-bit-Parallelschnittstelle ausgewertet werden kann. Mit diesem Interface und einem handelsüblichen Kassettengerät erhält man ein preiswertes, mobiles Aufzeichnungssystem, das zusammen mit einem BASIC-Tischrechner (z.B. COMMODORE CBM 3032) ein ideales Hilfsmittel für viele meßtechnische Aufgaben ist.

Das Interface besteht aus einem unabhängigen **Aufnahme-** und **Wiedergabesystem** sowie einer **Start-/Stopp-Einrichtung** für das Tonbandgerät bei der Aufnahme.

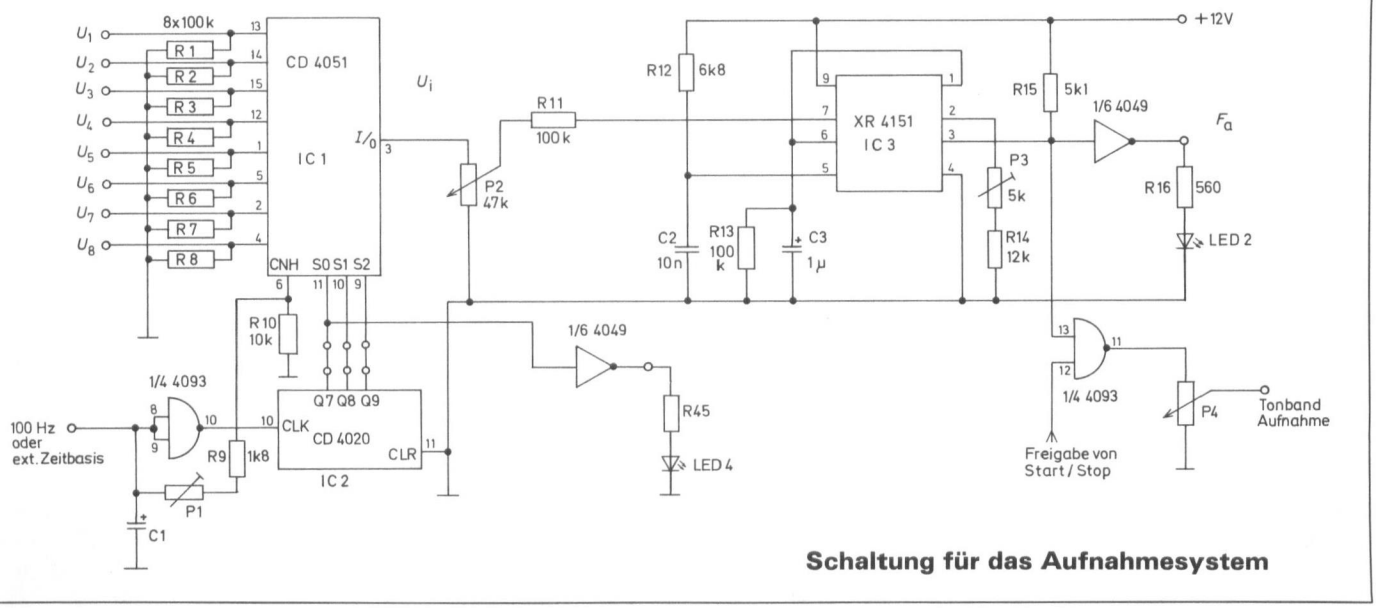
## Das Aufnahmesystem

Das *Analogdaten-Aufnahmesystem* besteht aus einem 8-Kanal-Analogmultiplexer, einem Spannungs-/Frequenzwandler und einem NF-Modulator.

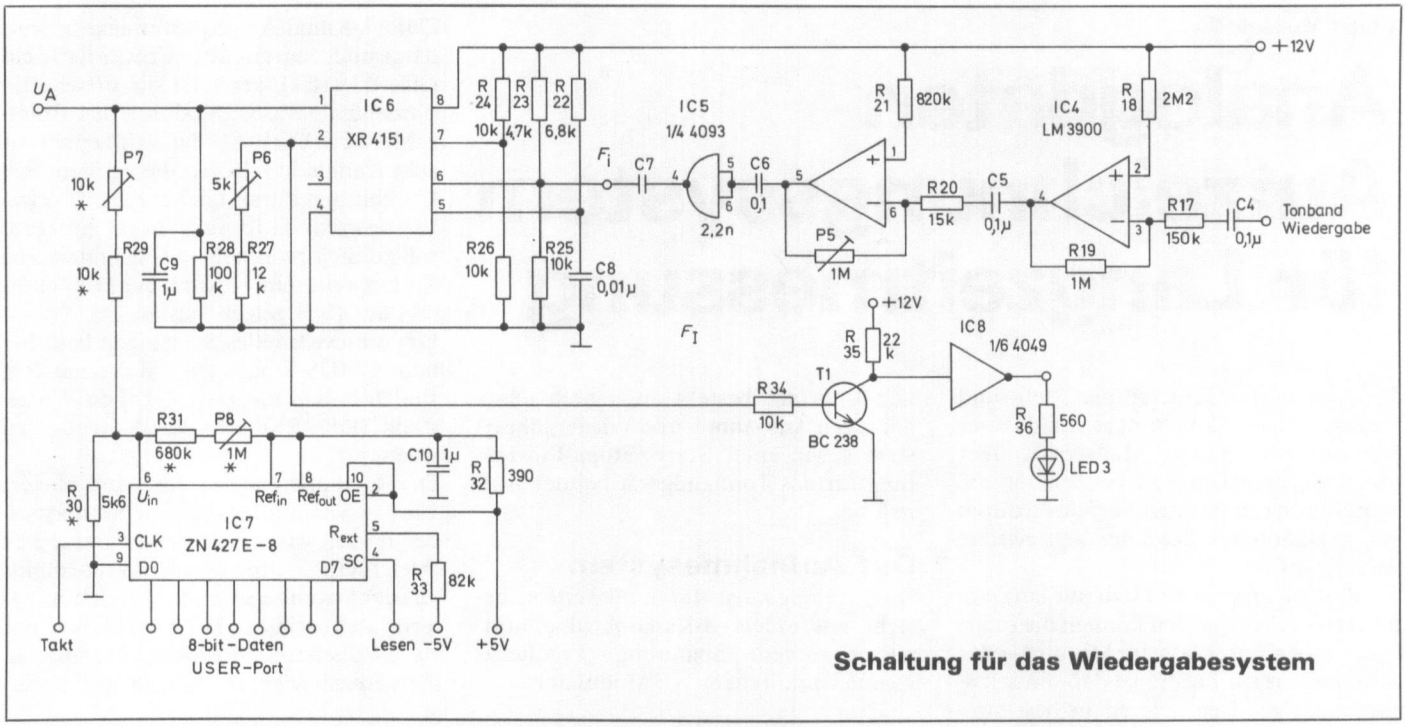
Der 8-Kanal-Analogmultiplexer wird dargestellt durch den CMOS-Baustein CD4051 (IC1), der über die drei Selektionssignale S0, S1, S2 durch den Binärzähler IC2 (CD4040 oder 4020) einen der acht Kanäle I1 – I8 auf die gemeinsame I/0-Leitung durchschaltet. Der Freigabeeingang INH ist auf L-Pegel zu legen; soll zeitlich zwischen zwei Kanälen eine Trennpause sein, so kann der Oszillatortakt auf INH gelegt werden.
Um ein eventuelles Schwingen bei offenen CMOS-Eingängen zu vermeiden, sind U1 – U8 mit einem hohen Widerstand (R1 – R8) auf Massepotential zu legen.
Die Umlauffrequenz des Multiplexers kann – um eine spätere Synchronisation bei der softwaremäßigen Auswertung zu erleichtern – über einen 100-Hz-Impuls aus der Stromversorgung, durch eine externe Zeitbasis der bei eingesetzten R9, P1, C1 über einen freilaufenden Oszillator geregelt werden. Wegen der Auflö-




**Blockschaltbild**




**Schaltung für das Aufnahmesystem**

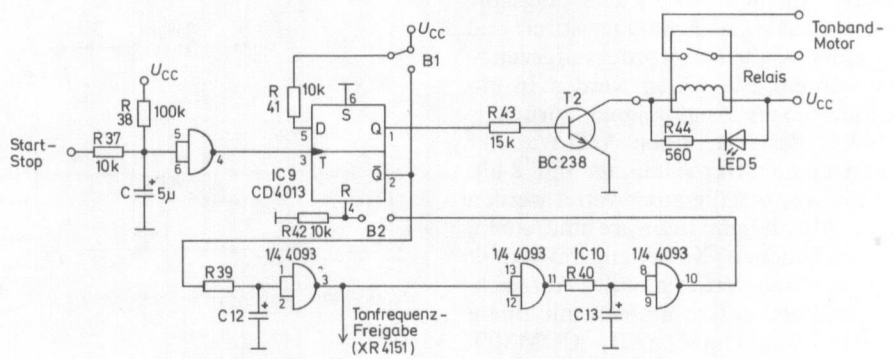



**Schaltung für das Wiedergabesystem**

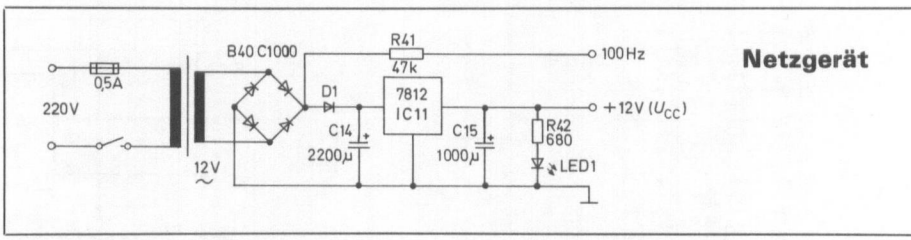
sung des UF-Wandlers ist die Umlauffrequenz am 4051 nicht kleiner als 1 s zu wählen, d.h., am Binärzähler (IC2) sind entsprechende drei aufeinanderfolgende Q-Ausgänge als Selektionssignale zu benutzen. Man erhält z.B. bei 100-Hz-Takt und mit Q7, Q8, Q9 eine brauchbare Umlauffrequenz von 128 : 100, d.h. etwa 1,3 s.

Mit dem Einsteller P2, der als Spannungsteiler wirkt, können Analogspannungen über 10 V um einen konstanten Faktor heruntergesetzt werden.

Der Hauptbestandteil des Aufnahmeteiles ist ein Spannungs-/Frequenzwandler mit dem integrierten Baustein XR 4151 (IC3). Dieser erzeugt mit geringem Aufwand an passiven Bauelementen aus einer Eingangsspannung $U_i$ von 0 bis 10 V eine proportionale Ausgangsfrequenz $f_a$ von genau 0 bis 10 kHz mit einer Linearität von +1%. Der Einsteller P3 dient hierbei zum Abgleich. Die Aufnahmetonfrequenz wird über ein Start-/Stopp-Tor freigegeben. Die Amplitude kann durch den Regler P4 vor der Aufzeichnung eingestellt werden.




**Schaltung der Start-/Stopp-Einrichtung**




**Netzgerät**

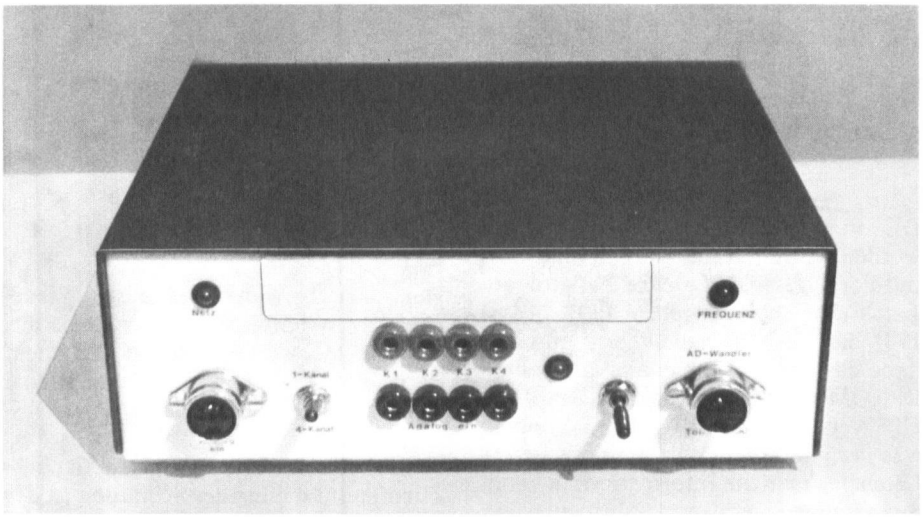
## Das Wiedergabesystem

Das Wiedergabesystem für die vorher auf Band aufgezeichneten Analogdaten besteht aus einem *NF-Tonfrequenz-Demodulator,* einem *Frequenz-/Spannungswandler* und einem *Analog-/Digitalwandler.*

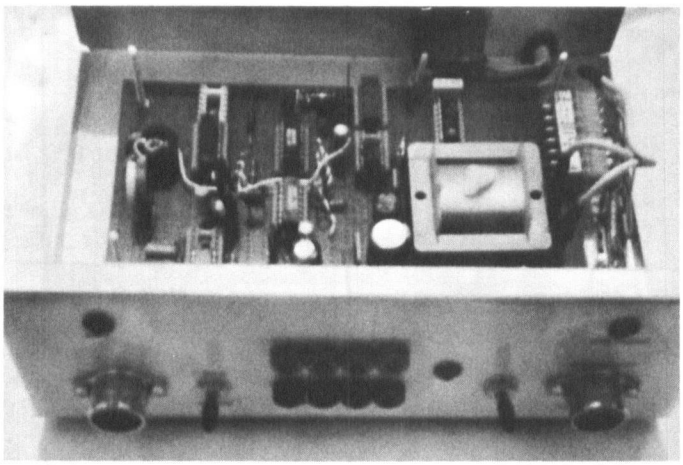
Der *Nf-Demodulator* hat die Aufgabe, das schwache Tonsignal in ein sauberes CMOS-kompatibles Rechteck umzuformen. Dies wird realisiert durch einen zweistufigen Wechselspannungsverstärker mit dem Operationsverstärker LM 3900 (IC4). Über den Einsteller P5 kann der Verstärkungsfaktor justiert werden. Der nachgeschaltete Schmitt-Trigger stellt ein sauberes Rechtecksignal bereit.

Der *Frequenz-/Spannungswandler* arbeitet ebenfalls mit dem integrierten Baustein XR 4151 (IC6) in der umgekehrten Betriebsart. Die Eingangsfrequenz $F_i$ zwischen 0 bis 10 kHz wird in eine proportionale Spannung $U_a$ von 0 bis 10 V umgewandelt. Der Einsteller P6 dient wiederum zum Abgleich der Ausgangsspannung. An dieser Stelle erhält man das rekonstruierte gemultiplexte Analogsignal.

Dieses wird über einen einfachen 8-bit-A/D-Wandler ZN427E-8 (IC7) in eine für den Rechner verständliche 8-bit-Digitalformation umgewandelt, die beim




**Aufgebautes Gerät für ein Aufzeichnungssystem**



**Geöffnetes Gerät mit der Schaltung für das Aufnahmesystem**

Anschluß an einen Commodore-Rechner über den USER-Port eingelesen wird. Mit P7 wird die maximale Analog-Amplitude A mit P8 der Nullabgleich eingestellt.
Bei der Wahl des Wandlungstaktes kann man der Einfachheit halber das auf dem USER-Port liegende TV-Horizontal-Signal von etwa 15 kHz verwenden. Da es sich um langsame Analogprozesse handelt, ist eine Wandlungsrate von etwa 1 ms ≙ 18 Takte (≈60 µs Taktdauer) in jedem Falle ausreichend. Der Lesevorgang wird durch einen L-Impuls am Ausgang (CB2) eingeleitet, der das Start-of-Convertion-Signal (SC) bildet. Da der Lesebefehl PEEK (USER Port) ca. 50 ms benötigt, braucht das End-of-Conversion-Signal (EC) nicht abgewartet zu werden.

## Die Start-/Stopp-Einrichtung

Um eine sinnvolle Aufzeichnung mit optimal ausgenutztem Band zu erhalten, ist es bei weit auseinanderliegenden Meßintervallen notwendig, das Bandgerät über eine automatische Start-/Stopp-Schaltung im richtigen Moment ein- bzw. auszuschalten.
Hierbei kann das auslösende Signal z.B. eine manuell bediente Taste oder ein Impuls einer externen Zeitbasis sein. Es sind zwei Betriebsarten denkbar:

- Es gibt einen Start- und einen Stopp-Impuls
- Es gibt nur einen Start-Impuls, die Abschaltung erfolgt automatisch nach einer bestimmten Zeit.

Es ist zu beachten, daß der Beginn der Datenaufzeichnung erst kurz nach dem Einschaltvorgang des Kassettenmotors erfolgen darf. Dies wird durch die Schaltung in Bild 4 – bestehend aus einem Flipflop CD 4013 (IC9), einem Verzögerungsglied und einem Monoflop (IC 10) – realisiert.
C11, R37, R38 dienen zur Tasten-Entprellung des Start-Signals. Nach der negativen Flanke schaltet der Q-Ausgang des Flipflops über den Transistorverstärker und das Relais den Motor ein. Nach einer kurzen Verzögerungszeit t1 (C12, R39) wird das Tonfrequenzsignal freigegeben und das Monoflop gestartet, dessen Zeitdauer t2 durch C13 und R40 festgelegt ist.
In der Betriebsart „Start/Stopp" wird der $\bar{Q}$-Ausgang des Flipflops über einen Widerstand auf den D-Eingang gebrückt, d.h., das Flipflop ist als Teiler durch 2 geschaltet (Brücke B1).
In der zweiten Betriebsart wird der Ausgang des Monoflops auf den RESET-Eingang des Flipflops geschaltet (Brücke B2).

## Stromversorgung

Eine einfache 12-V-Stromversorgung ist für die gesamte Schaltung ausreichend. Die zur Versorgung des A/D-Wandlers (IC7) erforderliche Versorgungsspannung von +5 V und −5 V werden nur bei der Auswertung durch den Rechner benötigt. Um weiteren Hardwareaufwand zu vermeiden, können diese aus dem Commodore-Rechner zum Interface herausgeführt werden.
Um eine Toleranz von 1% einzuhalten, sind die mit * gekennzeichneten passiven Bauelemente mit einer entsprechenden Toleranz zu wählen.

## Aufbau

Folgende Signale werden nach außen geführt:

- Aufnahme-/Wiedergabe-Leitung zum Tonband
  (z.B. über 5polige Standard-DIN-Buchse)
- Acht Analog-Eingabe-Kanäle (z.B. über acht Paar Telefonbuchsen)
- 220-V-Anschluß bzw. potentialfreie Schließerkontakte für Tonbandmotor
  (z.B. über Schuko-Steckdose oder zwei Telefonbuchsen)
- Start-/Stopp-Signal
- Acht Datenbit, Steuersignal und 0 V für USER-BUS-Anschluß (z.B. über 15poligen Cannon-Stecker)

Folgende Signale werden zur Kontrolle durch LED dargestellt:

1. Versorgungsspannung
2. Aufnahmefrequenz
3. Wiedergabefrequenz
4. Multiplex-Frequenz
5. Start-/Stopp-Flipflop

Christoph Pape

# Schrift und Grafik aus einem IC

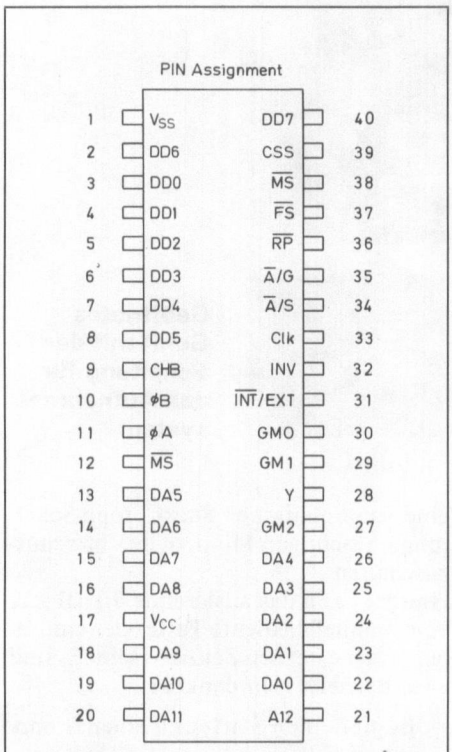
Ein einzelner Baustein übernimmt die gesamte Steuerung für ein Fernseh-Interface mit Grafik. Dies ist der VDG (Video-Display-Generator) MC 6847 von Motorola.
Im Alpha-Modus ist zwar die Auflösung von 16 Linien à 32 Zeichen nicht gerade üppig, aber es reicht durchaus, wenn man nicht gerade ein komfortables Textverarbeitungssystem braucht.
Der Umstand, daß der VDG ein Signal nach der amerikanischen Norm erzeugt, bringt einige kleine Einschränkungen:

- Es ist kein Farbbetrieb nach der PAL-Norm möglich. Das spielt in den meisten Fällen keine Rolle, da die meisten Hobbycomputerfans nur über einen Schwarzweißmonitor/-fernseher verfügen.
- Es können Interferenzen mit unserer Netzfrequenz auftreten, da die Halbbildwechselfrequenz 60 Hz beträgt. Außerdem können Synchronisationsschwierigkeiten auftreten. Auf dem umgebauten Billigst-TV-Gerät, das ich als Monitor benutze, bekam ich jedoch ein gutes Bild.

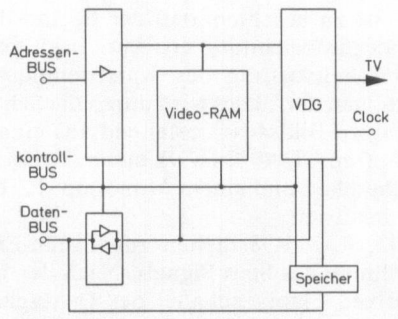
## Die Funktion im Computer

Das Blockschaltbild gibt Auskunft über die Funktion. Der VDG liest zyklisch das Video-RAM aus und stellt den Inhalt je nach Modus auf dem Bildschirm dar. Vom Computer aus arbeitet das Video-RAM wie ein normales RAM (Lesen/Schreiben).
Über die Adressen- und Datenbusbuffer (Tristate) ist dies möglich. Damit sich CPU und VDG nicht in die Quere kommen, werden, wenn die CPU auf die RAM zugreift, die Adressenausgänge des VDG hochohmig geschaltet (über $\overline{MS}$). Das hat den Vorteil, daß die CPU während ihrer Programmausführung nicht gestört wird. Leider hat diese Schaltung auch einen Nachteil. Wenn nämlich der VDG gerade Daten einliest, wenn die CPU das Video-RAM adressiert, kann ein sog. „Flicker" auf dem Bildschirm erscheinen, das heißt, es erscheint während ca. 0,5 bis 1 µs (je nach Zugriffzeit des Prozessors) ein Strich auf dem Bildschirm (ca. 0,5 bis 1 cm lang und eine Elektronenstrahlbreite hoch). Abhilfe würde hier eine Synchronisation zwischen CPU und VDG schaffen, die jedoch nur die CPU verlangsamen würde und zusätzliche Hardware benötigen würde.

Die Eingänge für den Betriebsmodus werden entweder über vorhandene Output-Ports geschaltet, oder es wird ein Latch (Speicher) eingebaut (z.B. 74(LS) 174), das diese Aufgabe übernimmt. Ich habe das Latch adressenmäßig in einen EPROM-Bereich gelegt, d.h., wenn die CPU versucht, ins EPROM zu schreiben, was ja nicht möglich ist, schreibt sie ins Latch. Da man die Information im Latch nicht mehr lesen kann, muß man die Information noch irgendwo im RAM speichern.




**Pinbelegung: MC 6847**




**Blockschaltbild**

## Das Herz der Schaltung: MC 6847

Der VDG übernimmt die gesamte Steuerung des Systems. Er liest die Daten aus dem RAM, dekodiert sie je nach dem Betriebsmodus und setzt das Videosignal zusammen. Die verschiedenen Betriebsmodis sind gezeigt. Je nach Modus werden 0,5- bis 6 KByte-RAM benötigt.
Zur Vereinfachung der Schaltung ist der $\overline{INT}$/EXT-Eingang auf 0 V (Masse) gelegt. Das bewirkt, daß der Modus Alpha mit externem Charakter-Generator und der Semigrafikmodus mit 64 × 48 Bildpunkten nicht benützt werden.
Das Signal Ā/S an den Datenbus D7 und INV an D6 geschaltet. Das bewirkt folgendes:

- Der Grafikmodus (Ā/G auf 0 V) ist uneingeschränkt benutzbar.
- Im Alphamodus werden sämtliche 256 Möglichkeiten eines Byte ausgenützt:
  Wenn D7 auf 0-Signal ist, werden die 64 Charakter des internen ROM dargestellt. Dabei geben die sechs niederwertigsten Bit (D0 – D5) den Buchstaben an (64 Charakter, 6-bit-ASCII). D6 bestimmt, ob der Buchstabe invertiert ist oder nicht (D6 = 0: normal, D6 = 1: invertiert).
  Wenn D7 auf 1-Signal ist, wird jedes Feld, auf dem sonst ein Buchstabe erscheint, in vier kleine Felder unterteilt (2 × 2). Dabei steuert das Byte, in dem sonst der Kode für den Buchstaben an dieser Stelle ist, dieses Feld, und zwar folgendermaßen:
  D0 – D3 bestimmt, ob das entsprechende Feld ON oder OFF ist, und D4 – D6 gibt an, welche Farbe diese Teilfelder haben.

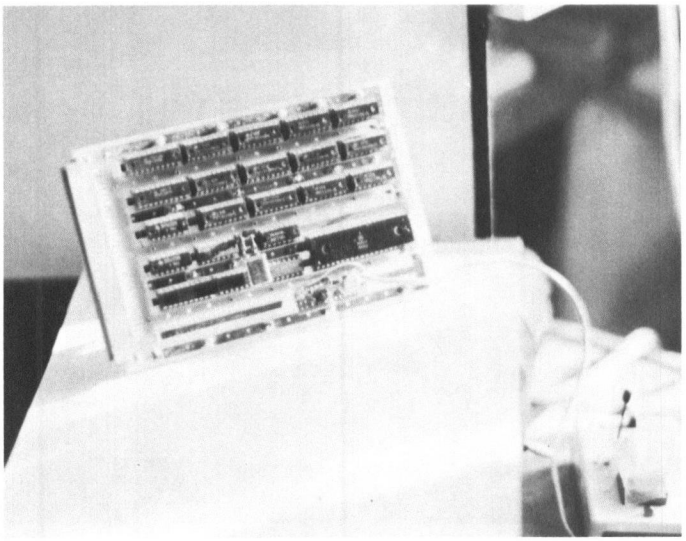
| D3 | D2 |
|---|---|
| D1 | D0 |

Ein Feld

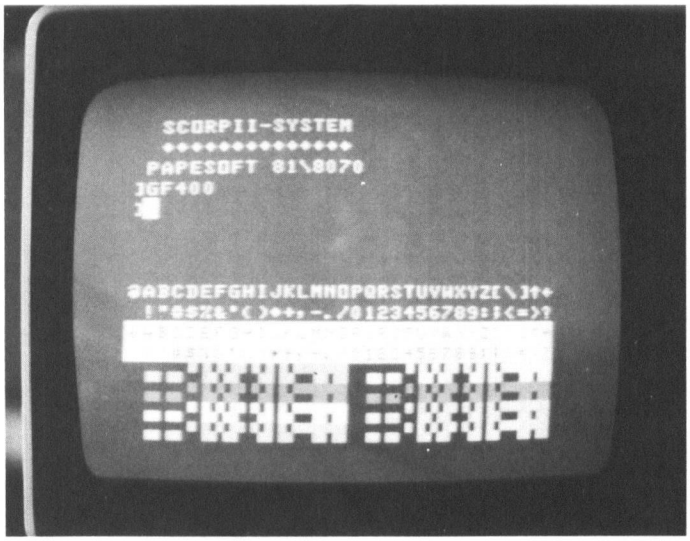
Ein Byte:

| D7 | D6 D5 D4 | D3 D2 D1 DO |
|---|---|---|
| 1 | Farbe | ON oder OFF |

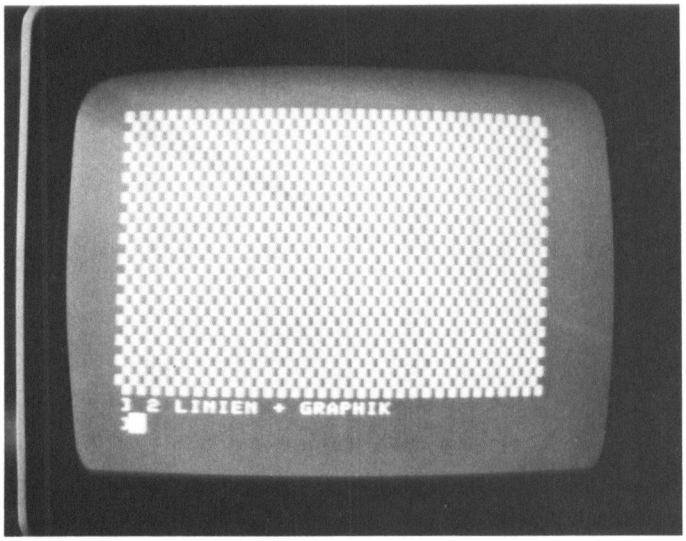
Die praktische Ausführung zeigt das Schema. Der Adressenbereich des Video-RAM ist 0800H – 1FFFH. Das Latch wird im Bereich 0000H – 07FFH dekodiert und zwar nur während eines Schreibzyklus. Dieser Bereich kann bei



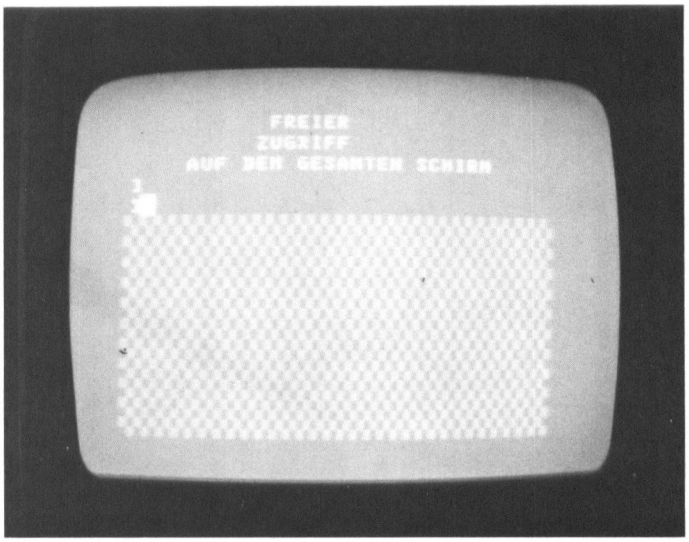
**Platine mit dem MC 6847 und den Steuereinheiten**



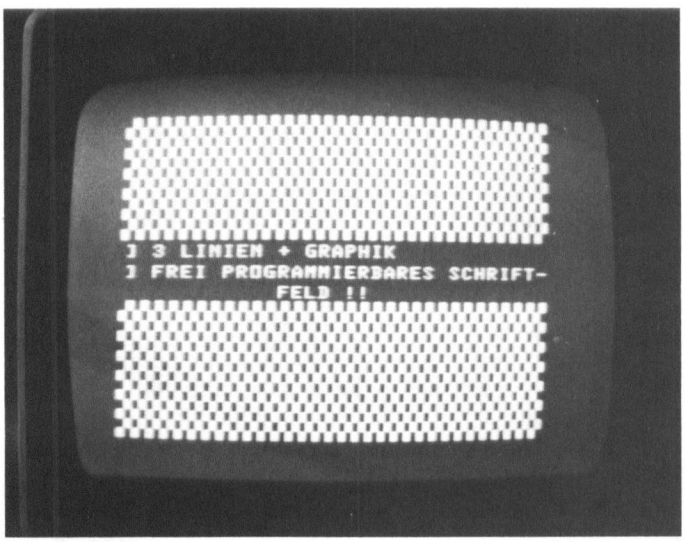

**Möglichkeiten der Darstellungen auf einem Bildschirm**



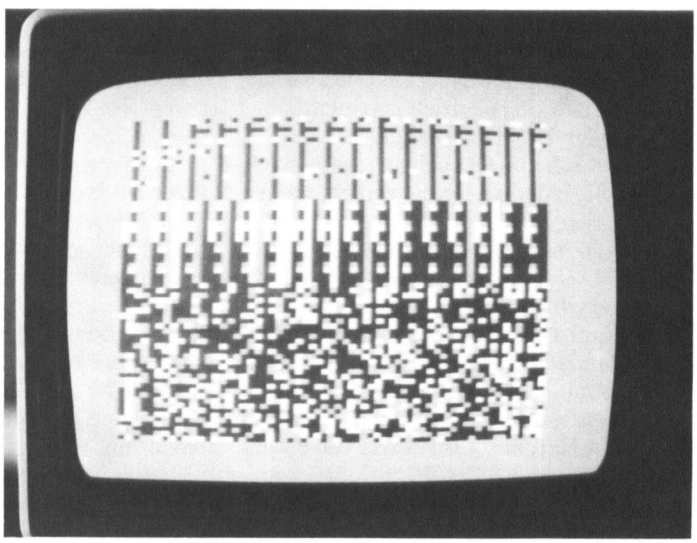

**Schachbrettmuster auf dem Bildschirm**



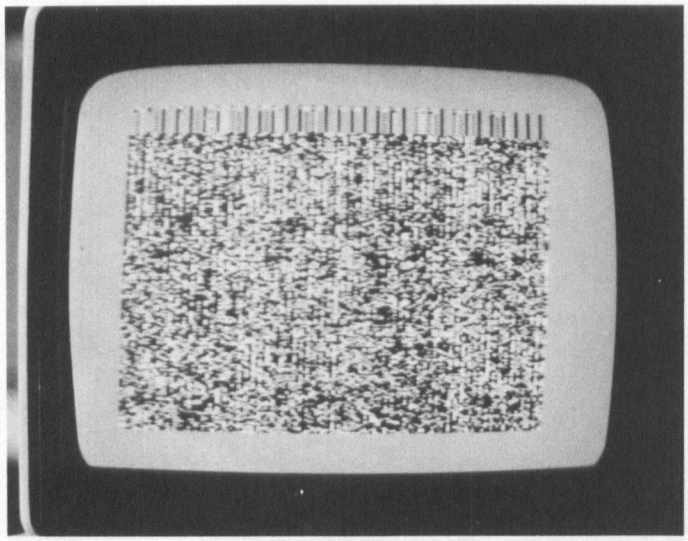

**Schachbrettmuster mit Cursor**



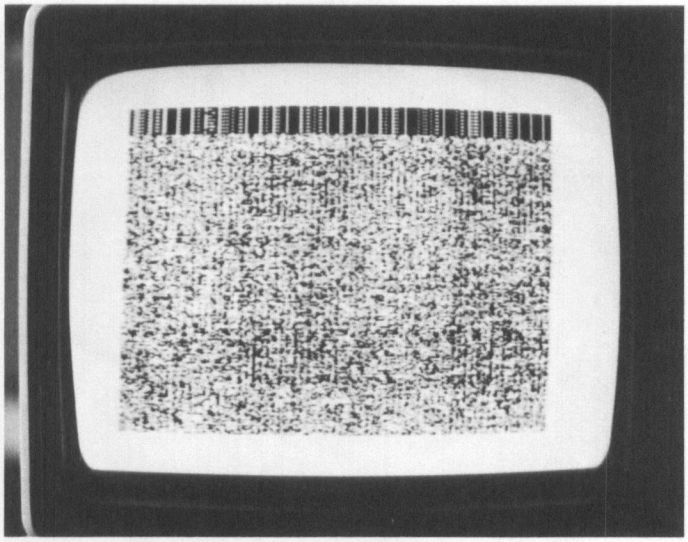

**Möglichkeiten zur Verschiebung und Ausblendung**



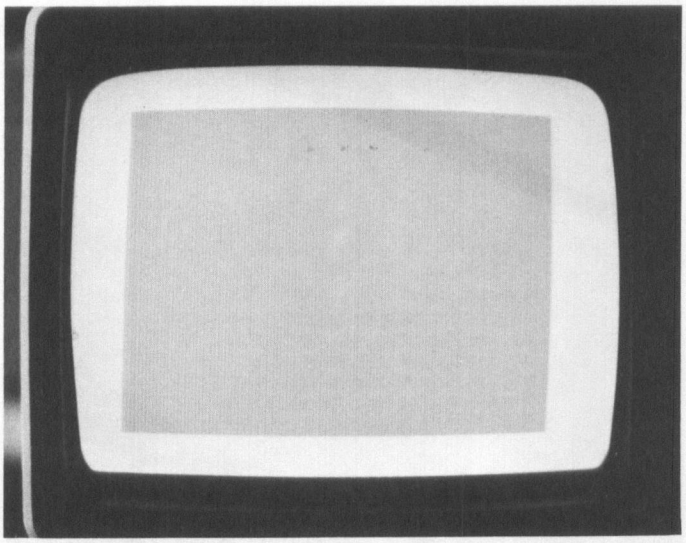
**Programmierbare Bildmuster auf dem Fernseher**



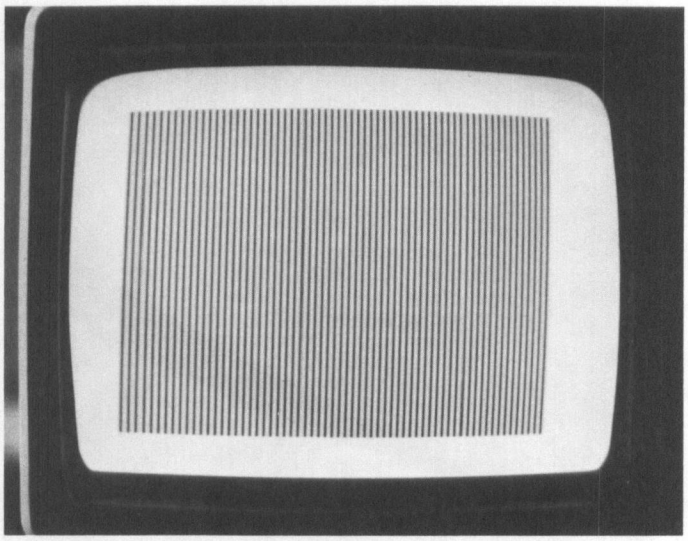
**Auflösung von programmierbaren Bildmustern**



**Weitere Hochauflösung von Bildmustern**



**Punktauflösung mittels des Bausteines MC 6847**

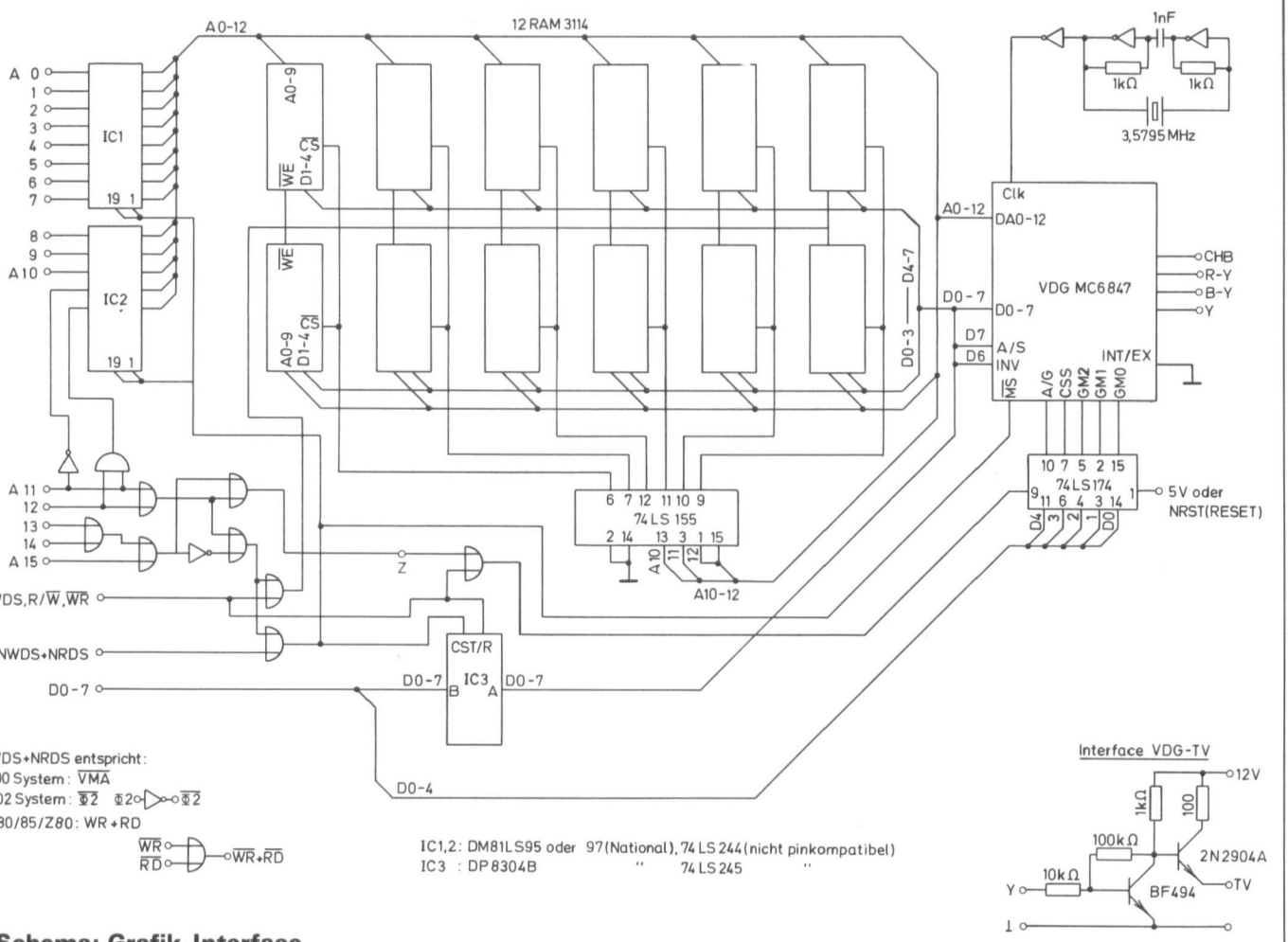


**Einfache Punktauflösung mit hell-dunkel-Quadraten**

anderen Mikroprozessoren ungünstig sein, der 6502 hat in diesem Bereich „Zeropage" und „Stack". Aus diesem Grund kann man am Punkt Z auf dem Schema anstatt mit dem Ausgang des ODER-Gatters mit dem $\overline{CS}$ (Chipselekt) irgendeines ROM oder EPROM verbinden.
Auf dem Schema ist auch angegeben, wie das Interface an die verschiedenen Mikroprozessoren angepaßt wird. Adressen- und Datenbus ist kein Problem. Das Schreib-Lese-Signal heißt zwar verschieden, ist aber in der Funktion identisch (NWDS, R/$\overline{W}$, $\overline{WR}$). Das letzte Signal zeigt an, daß die Adressen stabil sind und jetzt gelesen oder geschrieben wird. Bei meiner CPU, INS 8070 von National (in Prinzip ein erweiterter INS 8060 „SC/MP", mit Stackoperationen, 16-bit-Arithmetik, Direkt-Page-Adressierung und absoluten und indirekten Unterprogrammaufruf) gewinne ich dieses Signal aus NWDS und NRDS mittels ODER-Verknüpfung. Bei 8080/85/Z80 geschieht dasselbe mit den identischen Signalen $\overline{WR}$ und $\overline{RD}$. Beim 6800 ist es noch einfacher: dieses Signal entspricht $\overline{VMA}$. Auch der 6502 stellt ein solches Signal zur Verfügung, I2, das noch invertiert werden muß.
Über ein kleines Interface gelangt das Y-Signal, das die Schwarzweiß-Information und Synchronisationssignale beinhaltet, auf ein leicht modifiziertes TV-Gerät der billigsten Preisklasse.
Durch einen kleinen Eingriff erreichen wir einen besseren Kontrast auf dem Schirm, da ein HF-Modulator nur eine begrenzte Bandbreite übertragen kann.
Noch ein Wort zum Umbau: Beachten Sie, daß in einem TV-Gerät **lebensgefährliche** Spannungen vorhanden sind! Auch nach dem Ausschalten hat es noch geladene Hochspannungskondensatoren. Also einige Minuten warten, bis diese sich entladen haben.

## Die Software

Die unbeschränkte Mobilität, die man durch den VDG hat (freier Zugriff auf jeden Bildpunkt), setzt voraus, daß man für jede Bildschirmoperation ein Programm schreibt.
Beispiel: Cursorbewegungen mit blinkendem Cursor. Da muß das Programm den Cursor periodisch invertieren. Das setzt voraus, daß man irgendwo im Speicher die Position des Cursors speichert. Immer, wenn man ein Zeichen eingibt, z.B. über eine Tastatur, muß man das Zeichen an die betreffende Stelle setzen und die neue Cursorposition abspeichern. Danach muß geprüft werden, ob die neue Stelle überhaupt noch auf dem Schirm ist und gegebenenfalls ein Scrolling durchgeführt werden, d.h., alle Zeichen eine




**Schema: Grafik-Interface**

## Betriebsmodi

| Kontrollsignale | | | | | | | Anzeige | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Ā/G | Ā/S | GM 2 | GM 1 | GM 0 | CSS | INV | Farben | Auflösung | Bp/B | RAM |
| 0 | 0 | X | X | X | 0/1 | 0 | Alpha | 32 × 16 Char. | 1 | 512 |
| 0 | 0 | X | X | X | 0/1 | 1 | Alpha inv. | 32 × 16 Char. | 1 | 512 |
| 0 | 1 | X | X | X | X | X | 8 | 64 × 32 Char. | 4 | 512 |
| 1 | X | 0 | 0 | 0 | 0/1 | X | 4 | 64 × 64 Char. | 4 | 1024 |
| 1 | X | 0 | 0 | 1 | 0/1 | X | 2 | 128 × 64 Char. | 8 | 1024 |
| 1 | X | 0 | 1 | 0 | 0/1 | X | 4 | 128 × 64 Char. | 4 | 2048 |
| 1 | X | 0 | 1 | 1 | 0/1 | X | 2 | 128 × 96 Char. | 8 | 1536 |
| 1 | X | 1 | 0 | 0 | 0/1 | X | 4 | 128 × 96 Char. | 4 | 3072 |
| 1 | X | 1 | 0 | 1 | 0/1 | X | 2 | 128 × 192 Char. | 8 | 3072 |
| 1 | X | 1 | 1 | 0 | 0/1 | X | 4 | 128 × 192 Char. | 4 | 6144 |
| 1 | X | 1 | 1 | 1 | 0/1 | X | 2 | 256 × 192 Char. | 8 | 6144 |

Bp/B = Bildpunkte pro Byte

Zeile nach oben schieben. Ähnliches gilt für die Steuerfunktionen wie CR oder LF.
Doch diesen „Nachteilen", die einen Hobbyprogrammierer natürlich nicht erschüttern können, stehen gewaltige Vorteile gegenüber. So habe ich in einem Byte die Information, welches die oberste und unterste Linie ist. So kann ich softwaremäßig den Bildbereich einengen, um einen Teil des Schirmes für andere Zwecke zu benützen (Uhr, usw.).
Auch ist es möglich, direkt auf dem Bildschirm zu editieren und Zeichen vom Video-RAM wieder zurück in den Eingangsbuffer zu laden.
Für Anwender, die eine komfortable Textverarbeitung brauchen, ist die Kapazität vom 16 Linien à 32 Zeichen zu klein. Da jedoch die meisten Hobbycomputer zum Spielen gebraucht werden, ist der VDG mit seinen Grafikmöglichkeiten ideal (der neue TRS 80 mit seiner Farbgrafik benützt ebenfalls den MC 6847).

## Mögliche Erweiterungen

Wenn das Interface läuft, kommen natürlich Ideen, wie man das Interface erweitern könnte. Es wäre zum Beispiel zu prüfen, ob man mit den Farbausgängen (CHB, R-Y, B-Y) etwas machen kann. Eventuell kann man mit den Steuersignalen $\overline{RP}$, $\overline{HS}$, $\overline{FS}$ eine Logik steuern, um gemischte alpha- und hochauflösende Grafik darzustellen. Das läßt sich jedoch nur mit mehr Hardware erreichen, und die kostet Geld.

Frank Wähner

# Analog-Digital-Umsetzer

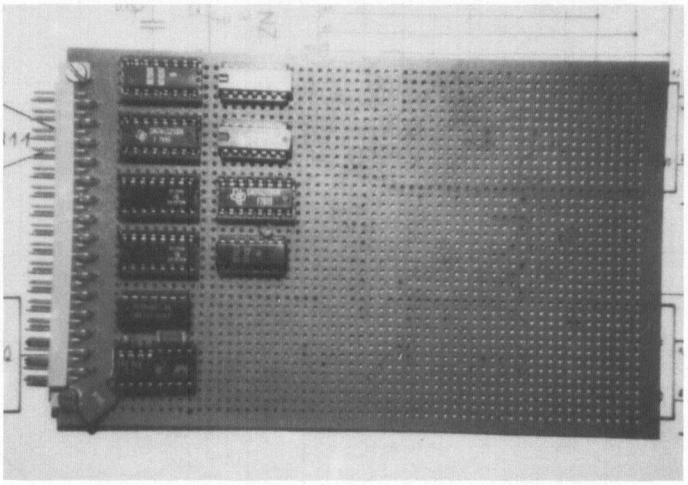
## Beschreibung des Beispielprogramms

### Zweck

Benutzung des CBM als Speicheroszilloskop

### Daten des CBM-Oszilloskops:

Eingangsempfindlichkeit: 1,28 $V_{SS}$ für vollen Ausschlag (ohne Verst.)
Frequenzgang: Gleichstrom bis max. 22 kHz (Abtastfrequenz: 45 kHz (N1 = Ø 1))
Triggerung: hängt vom Programm ab, hier ab einer Eingangsspannungsänderung >0,12 Volt
Anzeige: CBM-Bildschirm
Betriebsart: Speicheroszilloskop, d.h., Signal wird zuerst abgespeichert und dann angezeigt.



**Platine für den Analog-Digital-Umsetzer als ideale Ergänzung für Hobby-Computer**

### Eingabe des Programms

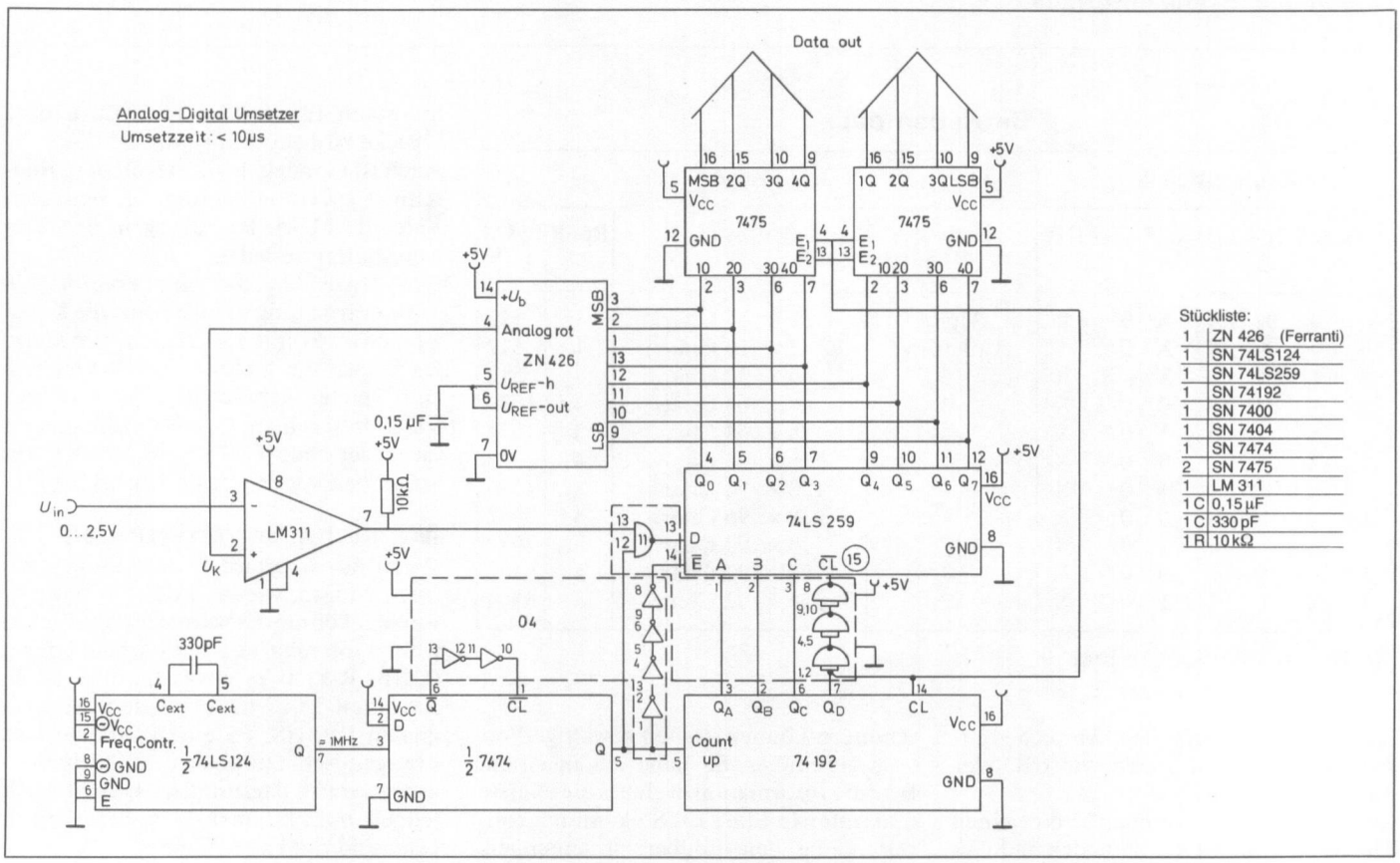
Mit Assembler des Verfassers ein Kinderspiel (siehe CHIP-SPECIAL 2, Programme 1980). Programm ist relozierbar. Startadresse beliebig. Das Programm arbeitet mit einem vom Verfasser in CHIP-SPEZIAL Programme II (dreidimensionale Darstellung auf 2001 CBM) veröffentlichten Unterprogramm zusammen (befindet sich im 2. Kassettenpuffer).

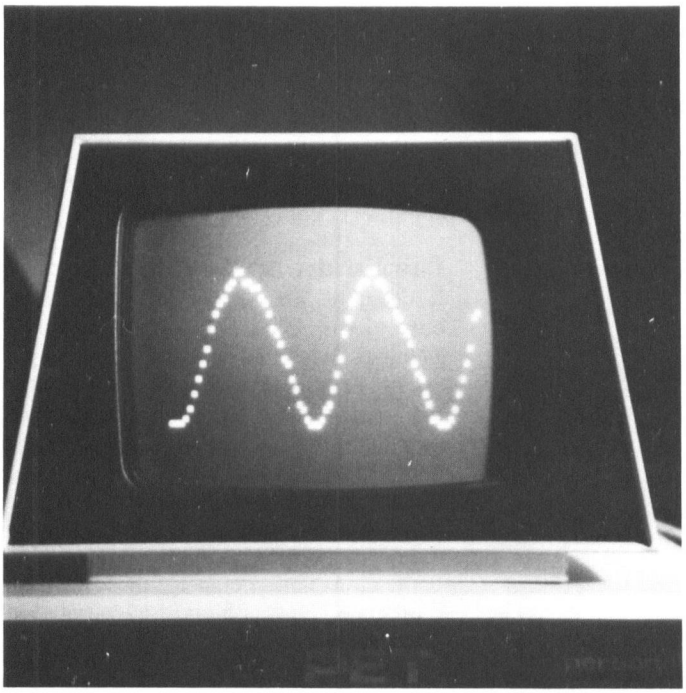
### Bedienung des Programms

Der A/D-Wandler wird an den Userport des CBM angeschlossen. Die Stromversorgung kann ebenfalls vom Computer genommen werden (5 Volt). Nach dem Starten des Programms arbeitet der CBM wie ein normales Oszilloskop. Die horizontale Ablenkfrequenz kann mittels der Variablen N1 eingestellt werden. Sie berechnet sich aus $F_H = 1000/(22 + (N1 - 1) * 5)$ [kHz]

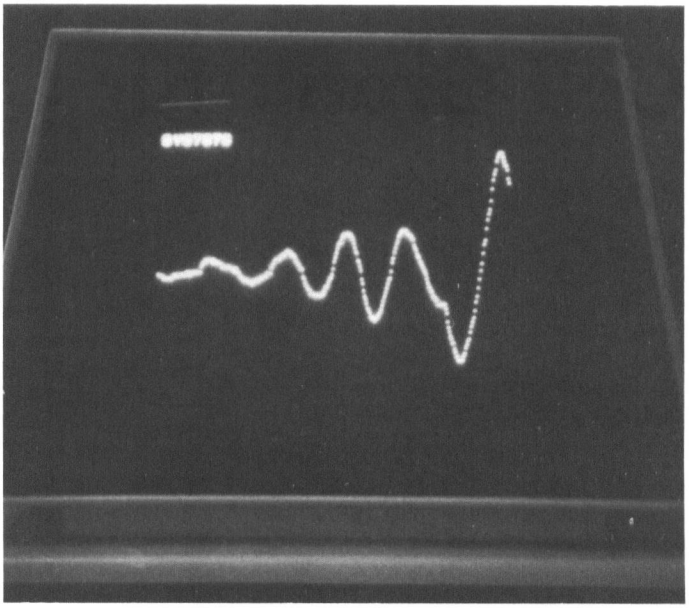
### Allgemeine Beschreibung

Das Programm dient eigentlich nur zur Demonstration der Leistungsfähigkeit





**Darstellung eines Tonsignals ≈ 500 Hz auf dem Bildschirm des CBM-Computers**



**Tonsignal ≈ 500 Hz**
**Mikroaufnahme: kurzes gesprochenes lautes „;", dargestellt auf einer Vollgrafikspeicher-Erweiterungs-Platine**

**Beispielprogramm für den Analog-Digital-Umsetzer für CBM:**

```
START   LDX #Ø
        STX 997      ; ZEILENINKREMENT AUF Ø
        STX 996      ; SPALTEN     "     "  "
        STX 994      ; PUNKT SETZEN
S1      LDX #Ø
        STX 992      ;SPALTE            AUF Ø
WAIT    LDA U
        STA 1
W1      JSR BREAK    ; STOP-TASTE GEDRUECKT ?
        SEC
        LDA U
        SBC 1
        BCS W2
        EOR #255
        ADC #1       ; BILDUNG DES KOMPLEMENT
W2      CMP #1Ø      ; SCHWELLENSPANNUNG ERREICHT ?
        BCC W1
        LDY #Ø
RECORD  LDA U
        STA TABLE,Y
DELAY   LDX #N1      ; N1 BESTIMMT ABTASTFREQENZ
D1      DEX
        BNE D1
        INY
        CPY #8Ø
        BCC RECORD   ; ABSPEICHERN FERTIG ?
        LDY #Ø
        LDA #128
        STY 1
        STA 2        ; INITIALISIEREN DES ZEIGERS
CLS     LDA #32
C1      STA (1),Y
        INY
        BNE C1
        INC 2
        LDA 2
        CMP #132
        BCC CLS      ; BILDSCHIRM GELÖSCHT ?
```

```
DRAW    LDY #Ø
DR1     LDA TABLE,Y
        EOR #255
        CLC
        ADC #1       ; BILDUNG DES KOMPLEMENT
        EOR #128
        STA 1
        AND #127
        LSR
        LSR
        LSR
        STA 2
        LSR
        ADC 2        ;VERKLEINERUNG
        LDX 1
        BMI NK       ; NEGATIVE SPANNUNG ?
        ADC #25
NK      STA 993
        JSR 826      ; PUNKT SETZEN
        LDY 992
        INY
        STY 992      ; SPALTE INKREMENTIEREN
        CPY #8Ø
        BCC DR1      ; FERTIG ?
        BCS S1       ; SPRUNG ZUR WAIT'ROUTINE
TABLE   .
        .
        .            ; 80 BYTES
        .
        .

U       = 59457      ; USERPORT (PORT A VON VIA #1)
BREAK   = 655Ø5      ; STOP-TASTE ABFRAGEROUTINE VON PET/CBM
```

des A/D-Umsetzers, dessen Geschwindigkeit vom CBM nur zur Hälfte ausgenutzt wird. Für ernsthaftere Anwendungen müßte man vor allem die Triggerung verbessern (Software). Ferner ist die Originalauflösung des CBM-Bildschirms wenig befriedigend. Auch könnte man einen vom Computer steuerbaren Vorverstärker vorsehen. Eine Anzeige des Frequenzspektrums des Eingangssignal wäre ein nicht sonderlich schweres Softwareproblem (FFT = Fast Fourrier-Transformation). Entsprechende bereits existierende BASIC-Programme müßten lediglich compiliert werden.

Stephan Hess

# Cell-Grafik für CBM

Die Cell-Grafik ermöglicht das Setzen oder Löschen eines beliebigen Punktes innerhalb einer Zeichenmatrix ähnlich einem Dot-Grafik-System (wie beim Apple), nur das statt 64 000 nur 64 Pixel gesteuert werden können. Allerdings kann man mehrere Zeichen definieren und jedes durch einen eigenen Kode bzw. eine Taste abrufen.

Bei dem vorliegenden Schaltungsvorschlag handelt es sich um eine 8 × 8-Matrix, und es können bis zu 128 Zeichen definiert werden.

Alle Cursor- und Steuerfunktionen einschließlich RVS und 2″ bleiben erhalten (das 2 gilt nur für den CBM, da bei anderen Geräten keine Steuerfunktion).
Die Cell-Grafiks können über Hard- oder Software ein- und ausgeschaltet werden.

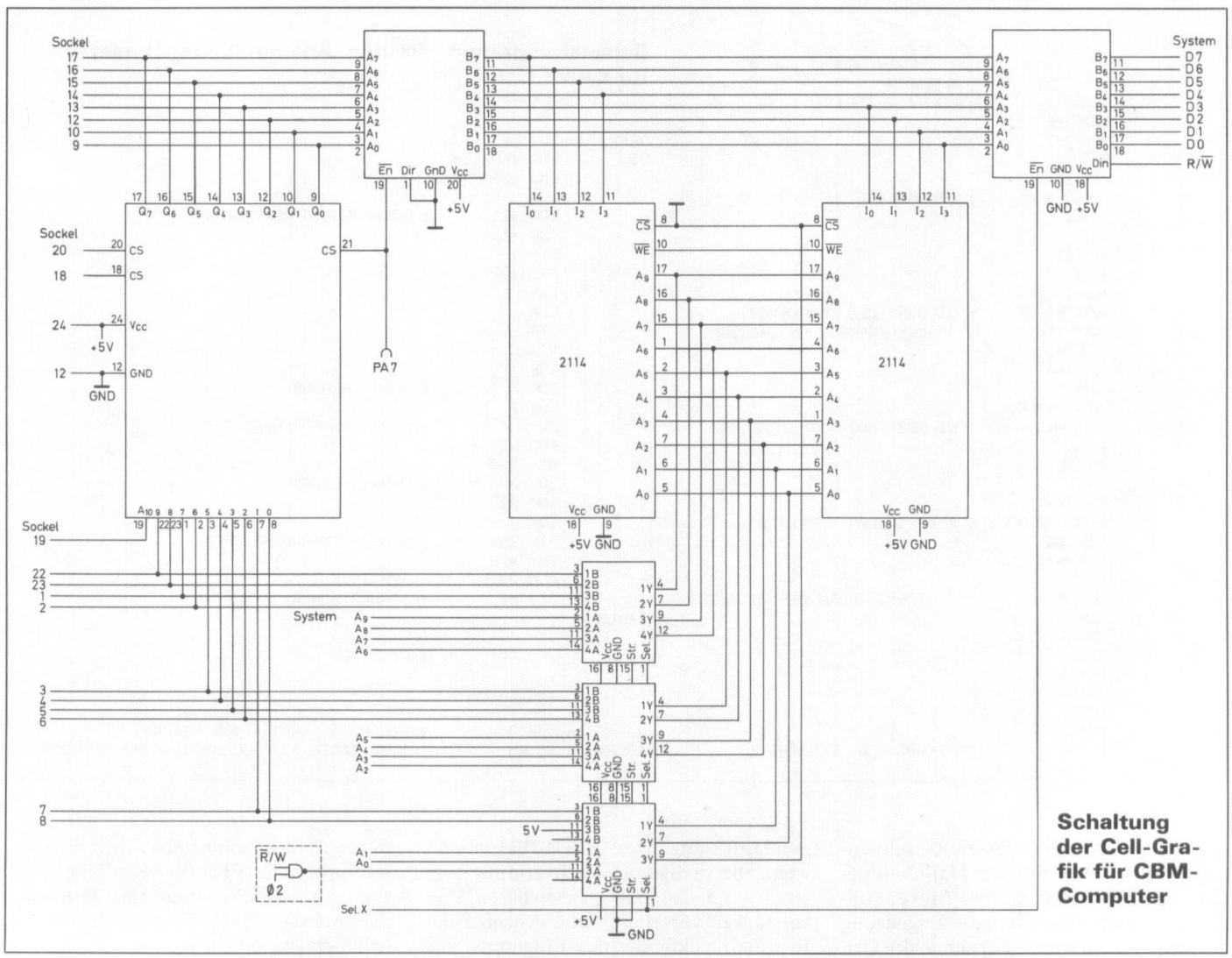
Die Zeichengenerierung erfolgt mit Hilfe eines Programms. Damit können Symbole von 1 × 1- bis 4 × 3-Matrizen bzw. 8 × 8- bis 32 × 24-Pixel erzeugt werden.

Der Zeichensatz kann für spätere Verwendung auf Kassette abgespeichert oder nach Umschalten auf Cell-Grafik von der Tastatur oder einem Programm abgerufen werden. Cell-Grafik ist für alle die gedacht, die ungewöhnliche oder verschiedene Zeichensätze benötigen: für Spiele, Hardwareentwurf, Mathematik, Fremdsprachen o.ä.

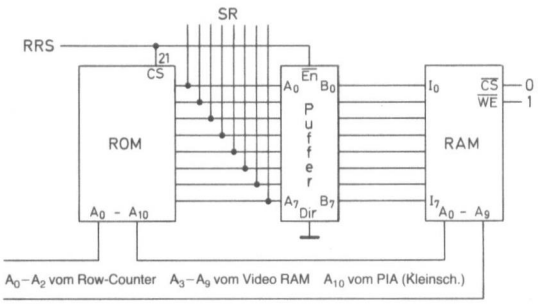
Wichtig: Ein Programm unterscheidet nicht zwischen den Zeichensätzen. Haben Sie ein Symbol auf der Taste A definiert und auf Cell-Grafik umgeschaltet, so behandelt das Programm das Symbol, als wäre es ein A, auch wenn etwas anderes auf dem Bildschirm erscheint (z.B. ein Integral oder ein Epsilon).

## Die Funktionsweise

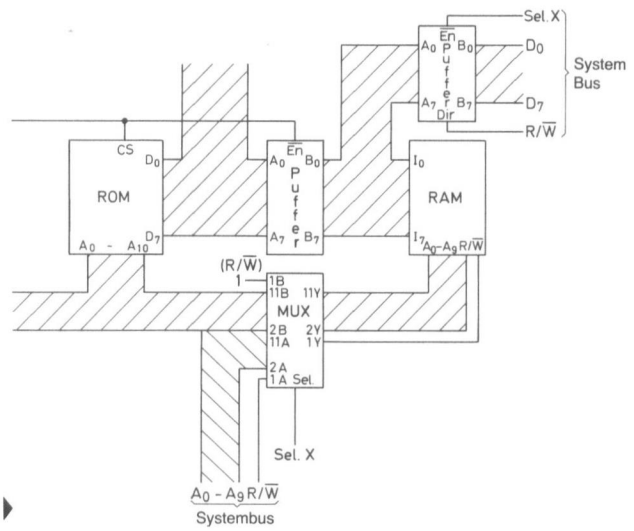
Ich beziehe alle Angaben auf den CBM, obwohl die Schaltung für alle Systeme geeignet ist, die einen Charaktergenerator besitzen und ihre Zeichen in einer 8 × 8-Matrix darstellen. Da der CBM kein Dot-Grafik-Display besitzt, muß ein Charaktergenerator vorhanden sein.
Dieser ist wie bei sehr vielen Systemen ein ROM. In diesem Falle ein EPROM vom Typ 2316B.
Will man den Zeichensatz erweitern, muß man dem schon vorhandenen ROM ein anderes parallel schalten und eine Umschaltmöglichkeit vorsehen. Will man den neuen Zeichensatz veränderbar gestalten, so wird einfach anstelle des zweiten ROM ein RAM verwendet.




**Schaltung der Cell-Grafik für CBM-Computer**




**Blockschaltbild mit ROM und RAM für die Cell-Grafik**




**Anschluß an die Busleitungen im CBM-Computer**

Als RAM verwendete ich zwei 2114. Dieses ermöglicht die Konstruktion von 128 Zeichen. Mehr Zeichen sind nicht sinnvoll, da von einem Byte nicht mehr adressiert werden können und das Bildschirm-RAM nur 8 bit an den Charaktergenerator übergibt. Soll sich der Zeichensatz ändern, so kann man ihn auch aus dem Hauptspeicher laden.

Wenn RAM und ROM parallel geschaltet werden sollen, müssen folgende Leitungen des RAM mit folgenden Anschlüssen verbunden werden:

Die Adressenleitungen mit den Adressenleitungen des ROM, und da in das RAM vom System her geschrieben werden muß, auch mit dem Adressenbus des Systems.

Die Datenleitungen mit dem Datenbus des ROM und ebenfalls mit dem System-Datenbus.

Die R/W-Leitung kann ständig an Read liegen, es sei denn, das System greift auf das RAM zu. Dann muß sie mit R/$\overline{W}$ verbunden sein.

Die CS-Leitung kann ständig auf Masse liegen, da die Datenleitungen über Tri-State-Buffer mit dem System verbunden sind.

Das ROM verfügt über drei CS-Eingänge, die vom Hersteller folgendermaßen programmiert wurden. Pin 18 und 20 auf L-Pegel, Pin 21 auf H-Pegel (gilt nur für den CBM).

Zuerst sollen die Datenleitungen gekoppelt werden.

Dazu werden die Datenleitungen des ROM mit den Ausgängen des Tri-State-Buffers 74LS245 verbunden.

Dessen Eingänge führen zu den Datenanschlüssen des RAM.

Da der Buffer durch einen L-Pegel an Pin 19 aktiviert wird, ist dieser Anschluß mit Pin 21 des ROM verbunden.

Dies bildet die Umschaltleitung für Cell-Grafik.

Sie wird nun mit einem Schalter oder, falls man Symbole verwenden soll, mit $PA_7$ verbunden.

Die Datenleitungen müssen noch mit dem System-Datenbus verbunden werden. Dazu wird ein weiterer Buffer mit den Datenanschlüssen des RAM verbunden. Die „andere" Seite des Buffers führt auf die System-Datenleitungen. Da der Datentransfer nur stattfinden darf wenn das RAM vom System angesprochen wird, ist Pin 19 des Buffers mit der entsprechenden Memoryselectleitung verbunden. Durch Hinzufügen eines 74LS42 kann Abhilfe geschaffen werden.

Die Richtung des Datentransfers wird durch System R/W bestimmt. Aus diesem Grunde wird R/W mit Pin 1 des Buffers verbunden. Da der Transfer bei L-Pegel zum RAM stattfindet, müssen die „A"-Anschlüsse mit dem RAM verbunden sein.

Da die Adressen, analog wie das R/W-Signal, für das RAM entweder vom System oder vom ROM kommen, sind diese beiden Signalquellen gemultiplext (74LS157). Die Auswahl der Signalquelle wird von der Memoryselectleitung gesteuert. Da diese auf L-Pegel ist, wenn das System auf das RAM zugreift, werden die Systemadressen A0 bis A9 mit den „A"-Eingängen der Multiplexer verbunden. Entsprechend die „B"-Eingänge mit den ROM-Adressen.

Die Ausgänge führen zu den entsprechenden RAM-Pin. Dasselbe gilt für die R/W-Leitung mit dem Unterschied, daß die R/W-Leitung vom ROM ein H-Pegel ist. Für 6502-Systeme kann es erforderlich sein, ein R/W-Signal zu erzeugen. Dies geschieht, indem man das Taktsignal $\Phi_2$ mit der invertierten R/W-Leitung mit NAND verknüpft.

## Der Umbau für andere Systeme

Die Cell-Grafik ist für alle Systeme mit Charaktergenerator und einer Zeichengeneratormatrix von 8 × 8 geeignet. Ausnahmen machen hier Systeme, bei denen Charaktergenerator und Schieberegister bzw. Multiplexer auf einem Chip integriert sind (z.B. CG 5004L-1) oder bei denen die achte Zeile mittels Hardware ausgeblendet wird. Bei einer 7 × 5-Zeichen-Matrix ist dies häufig der Fall. Auch kann es vorkommen, daß ein Eingang des Schieberegisters bzw. des Multiplexers mit Masse verbunden ist. In diesem Falle muß diese Verbindung aufgetrennt und mit dem entsprechenden Anschluß des RAM gekoppelt werden.

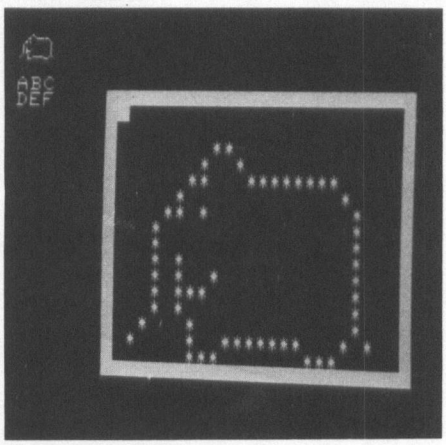
Nehmen wir einen günstigeren Fall an. Sie besitzen z.B. ein Video-Interface der Firma GES und wollen es umrüsten. Sie bauen das Gerät genau nach Schaltplan und ändern die Verbindungen zum ROM-Sockel entsprechend ab. Wie, das entnehmen Sie bitte dem Datenblatt des 2708.

Die CS-Leitung des 2708 ist auf L-Pegel, muß also noch einen Inverter vorgeschaltet bekommen.

## Die Software

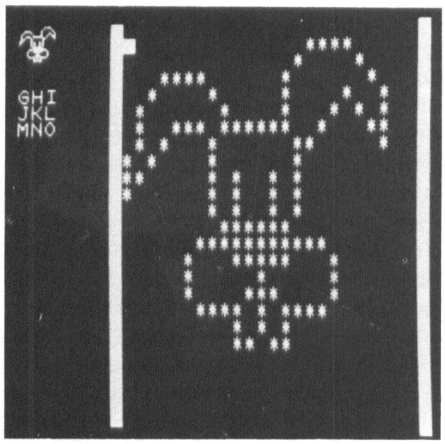
Das Definieren eines Zeichensatzes ist reichlich ermüdend. Aus diesem kühlen Grunde wurde das Programm „Symbole 1.1.17" geschrieben. Es ist für den CBM3032 gedacht. Im Assemblerlisting sind alle Adressen, die für die Verwendung auf anderen Ausbaustufen geändert werden müssen, unterstrichen. Die Änderung erfolgt, indem man aus 7X einfach 3X oder 1X macht. Zum Einge-

ben des Programms empfehle ich das Hexdumplisting. Erstens erhält es auch die nötigen Datenfelder, und zweitens sind garantiert alle Speicherzellen aufgelistet.
Das Programm wird mit G7100 gestartet. Daraus ist zu ersehen, daß sich das Programm im High-Memory (aus der Sicht von BASIC) befindet. Damit es nicht von Strings überschrieben wird, schützt es sich selbst durch Pointermanipulation. Auch wird der PIA initialisiert und $PA_7$ als Ausgang programmiert. Also wenn sie Symbole 1.1.17 verwenden wollen, sollten sie $PA_7$ zur RAM-ROM-Auswahl verwenden.
Nun denn, G7100 ist eingegeben und Return gedrückt. Der Bildschirm „blitzt" kurz auf, nicht um gleich darauf in zehntausend kleine Teile zu zerfallen, sondern um das „Menü" anzuzeigen. Der Druck einer Taste, die nicht dem Bereich A bis E entstammt, wird nicht beachtet.
Fangen wir unten an.
E:
Ende. Ich muß Sie enttäuschen, auch hier keine Selbstzerstörung. Nach Drücken der Taste „E" wird der definierte Zeichensatz in das Cell-Grafik-RAM übernommen, es wird auf „normalen" Zeichensatz umgeschaltet und in den Maschinenmonitor gesprungen.
D:
Erweiterung. Das Programm springt zu der in den Speicherzellen § 7163 (low) und § 7168 (high) abgespeicherten Adresse.
C:
C wie Zeichensatz. Das Programm zeigt den gesamten Zeichensatz in 1 × 1-Matrizen zerlegt an. Erschrecken Sie also nicht, wenn Sie ein Symbol, das als 2 × 3-Matrix definiert wurde, nicht auf Anhieb finden. Haben Sie sich satt gesehen, so drücken Sie eine beliebige Taste, und Sie werden sich im Menü wiederfinden.
A:
A wie Auch die Größe will definiert werden. Sie können nun Breite und Höhe Ihrer Zeichenmatrix definieren (Sie können in einem Zeichensatz beliebig wählen, d.h. maximal 12, da die Breite maximal 4, die Höhen maximal 3 sein kann. Geben Sie eine Zahl außerhalb des entsprechenden Bereichs von 1 bis 4 bzw. 1 bis 3 ein, so geschieht ... nichts. Eine Zeichenmatrix von 1 × 1 entspricht 8 × 8-Pixels, also genau der Größe eines „normalen" Zeichens. Definieren Sie eine größere Zeichenmatrix, so benötigen Sie später entsprechend mehr Tastendrücke, um das Symbol vollständig abzurufen.
Haben Sie Ihre Zeichenmatrix definiert, fragt Sie der Rechner nach einer Taste. Übrigens hätten Sie im Menü „B" gedrückt, so wären Sie an eben dieser Stelle gelandet. Die letzte eingegebene Felddefinition wäre dann immer noch gültig.
Der CBM verfügt über einige Tasten, also welche nehmen? Keine, drücken Sie lieber eine. Sollte die von Ihnen auserwählte eine Cursor-Kontroll-, die Return- oder sogar die BRK-Taste sein, so wird der CBM vor Scham den Bildschirm verdunkeln.
Das ist aber kein Grund zur Panik. Drükken Sie nämlich eine andere Taste, so wird der CBM Ihnen vergeben und das machen, was er, wenn Sie gleich eine „vernünftige" Taste gedrückt hätten, getan hätte.
... Verzeihung.
Links oben sehen Sie einen Elefant. Das Symbol ist in Originalgröße dargestellt.
Man hat nun seine Matrix definiert, die Taste gedrückt und will nun Zeichen definieren. Aber wie?
Drückt man eine Taste (außer Return, BRK, Home, Cursor rechts, Cursor Auf, Inst, Space, RVS, Shift, das wär's), so entsteht an der entsprechenden Stelle ein Asterics. Ebenfalls entsteht links oben im „Original" ein Punkt. Es handelt sich um eine Dauerfunktionstaste. Drücken Sie hingegen die Space-Taste, so wird entsprechend ein Punkt gelöscht.
Falls beim Punktesetzen eine Taste keinen Effekt zeigt, versuchen Sie eine andere.

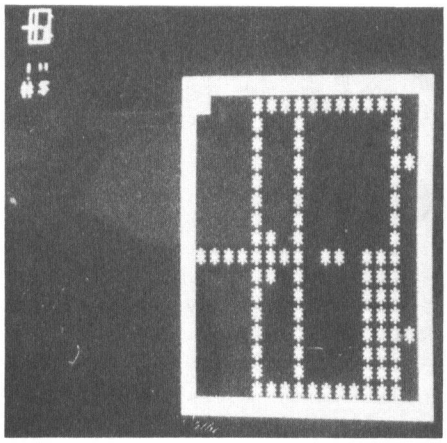



**Groß- und Klein-Darstellung eines Elefanten mit Hilfe der Cell-Grafik**

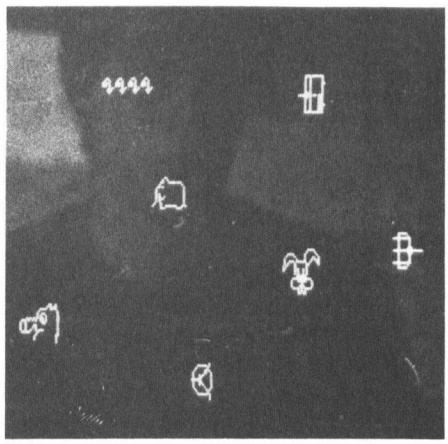
```
●● SYMBOLE 1.1.17 ●●

oooooooooo MENEUE oooooooooo

ADRESSE LABEL        OP-CODE OPERANT      1B 2B 3B

 7100                LDA   #$70           A9 70
 7102                STA    $53           85 53
 7104                LDY   #$00           A0 00
 7106                LDX   #$04           A2 04
 7108                STY    $FB           84 FB
 710A                STY    $FD           84 FD
 710C                LDA   #$7A           A9 7A
 710E                STA    $FC           85 FC
 7110                LDA   #$90           A9 90
 7112                STA    $FE           85 FE
 7114                LDA   ($FB),Y        B1 FB
 7116                STA   ($FD),Y        91 FD
 7118                DEY                  88
 7119                BNE   7114           D0 F9
 711B                INC    $FC           E6 FC
 711D                INC    $FE           E6 FE
 711F                DEX                  CA
 7120                BNE   7114           D0 F2
 7122                LDA   $E843          AD 43 E8
 7125                ORA   #$80           09 80
 7127                STA   $E843          8D 43 E8
 712A                LDA   $E841          AD 41 E8
 712D                AND   #$7F           29 7F
 712F                STA   $E841          8D 41 E8
 7132                LDA   $7E00,X        BD 00 7E
 7135                BEQ   713D           F0 06
 7137                JSR   $E3D8          20 D8 E3
 713A                INX                  E8
 713B                BNE   7132           D0 F5
 713D                STY    $9E           84 9E
 713F                LDA    $9E           A5 9E
 7141                BEQ   713F           F0 FC
 7143                LDA   $026F          AD 6F 02
 7146                CMP   #$41           C9 41
 7148
```




**Groß- und Klein-Darstellung eines Hasen auf einem CBM-Computer**



**Darstellung eines elektronischen Schaltsymboles durch die Cell-Grafik**



**Darstellungsmöglichkeiten der Cell-Grafik, wobei sich für den Anwender große Variationen ergeben**

!!!!!! Cursor-Bewegungen erfolgen nicht durch die Cursor-Kontrolltaste !!!!!!!!
Dazu werden die Zifferntasten 1 bis 9 verwendet.
Die Taste 5 löscht das Bild und setzt den Cursor in die linke obere Ecke.
Die anderen Zifferntasten verrücken den Cursor in die Richtung, in der sie sich (auf die 5 bezogen) befinden. Auch das sind Dauerfunktionstasten.
Shift, RVS und BRK haben keine Funktion, ebenso INST.
Home bringt Sie ins Menü zurück. Cursor ab hat die Frage nach der Taste zur Folge.
Return schaltet auf die nächste Matrix weiter. Falls das darauffolgende Zeichen in einer anderen Matrix definiert ist, können sich seltsame Bilder ergeben.
Cursor links schaltet auf die vorhergehende Matrix um.
Wollen Sie einen Zeichensatz auf Kassette speichern, so geben sie folgendes ein (im Maschinenmonitor):

S „name", 01, 7600, 7A00. Zum Zurückholen einfach 1 eingeben. Vorher muß Symbol schon geladen sein. Sie können es aber durch eine Speicherverschieberoutine in den Bereich 9000 bis 9400 transferieren.

```
                BCC   713D          90 F3
714A            CMP   #$46          C9 46
714C            BCS   713D          B0 EF
714E            AND   #$07          29 07
7150            TAX                 AA
7151            LDA   $715F,X       BD 5F 71
7154            STA   $715E         8D 5E 71
7157            LDA   $7164,X       BD 64 71
715A            STA   $715F         8D 5F 71
715D            JMP   $716A         4C 6A 71
7160            ***                 92
7161            INY                 C8
7162            AND    (00),Y       31 00
7164            ROR                 6A
7165            ADC    (71),Y       71 71
7167            ***                 74
7168            ADC    (71),Y       71 71

716A            LDX   #$04          A2 04
716C            LDY   #$00          A0 00
716E            STY    $FB          84 FB
7170            STY    $FD          84 FD
7172            LDA   #$76          A9 76
7174            STA    $FC          85 FC
7176            LDA   #$90          A9 90
7178            STA    $FE          85 FE
717A            LDA    (FB),Y       B1 FB
717C            STA    (FD),Y       91 FD
717E            DEY                 88
717F            BNE   717A          D0 F9
7181            INC    $FC          E6 FC
7183            INC    $FE          E6 FE
7185            DEX                 CA
7186            BNE   717A          D0 F2
7188            LDA   $E841         AD 41 E8
718B            ORA   #$80          09 80
718D            STA   $E841         8D 41 E8
7190            BRK                 00
7191            BRK                 00
```

## Umschalten auf Cell-Grafik

Die Umschaltung kann sowohl hard- als auch softwaremäßig erfolgen.

## Hardwareumschaltung

Man verbindet die Umschaltleitung mit einem Schalter, der an Masse liegt. Wird nun geschaltet, wird auch CS des ROM und Enable des Buffers auf Masse gezogen und damit auf Cell-Grafik umgeschaltet.
Dies ist auch möglich, wenn Sie $PA_7$ verwenden.
Sie müssen aber darauf achten, daß $PA_7$ als Eingang programmiert ist (POKE 59459,0).

## Softwareumschaltung

Dazu wird $PA_7$ als Ausgang programmiert (POKE 59459,128). Nun können Sie durch POKE 59457,128 bzw. POKE 59457,0 auf „normalen" oder „privaten" Zeichensatz umschalten. Ich empfehle Ihnen, die Space-Taste immer mit einem Leerfeld zu belegen, damit der Bildschirm mit Clr. Home gelöscht werden kann.
Das Umschalten kann natürlich auch aus einem Programm heraus erfolgen.

Klaus Kohlschütter

# Uhr/Timer/Kalender für µP-Systeme

Es gibt sicher viele kommerzielle Mikrocomputersysteme, in denen eine Uhr, evtl. mit Datumanzeige, integriert ist. Aber die vielen Mikrocomputer, die in Eigenbau hergestellt wurden, verfügen meistens nicht über eine solche Schaltung. Die Anwendbarkeit einer solchen Schaltung ist sehr groß, da sie für den Report auf Floppy-Disk-Systemen fast unentbehrlich ist, zumal wenn die Uhrzeit immer zur Verfügung steht, wie in der vorgestellten Schaltung, selbst dann, wenn der Mikrocomputer oder ein System ausgeschaltet ist.
Die vorgestellte Uhr geht über den Rahmen einer normalen digitalen Uhr weit hinaus, da diese Uhr nicht nur die Zeit angeben kann, sondern auch das Datum mit automatischer Umschaltung bei unterschiedlichen Monatslängen angibt und zudem noch als Timer oder Real-Time-Clock Verwendung finden kann.
Was sind die Vorzüge dieser Uhr/Timer/Kalender-IC:

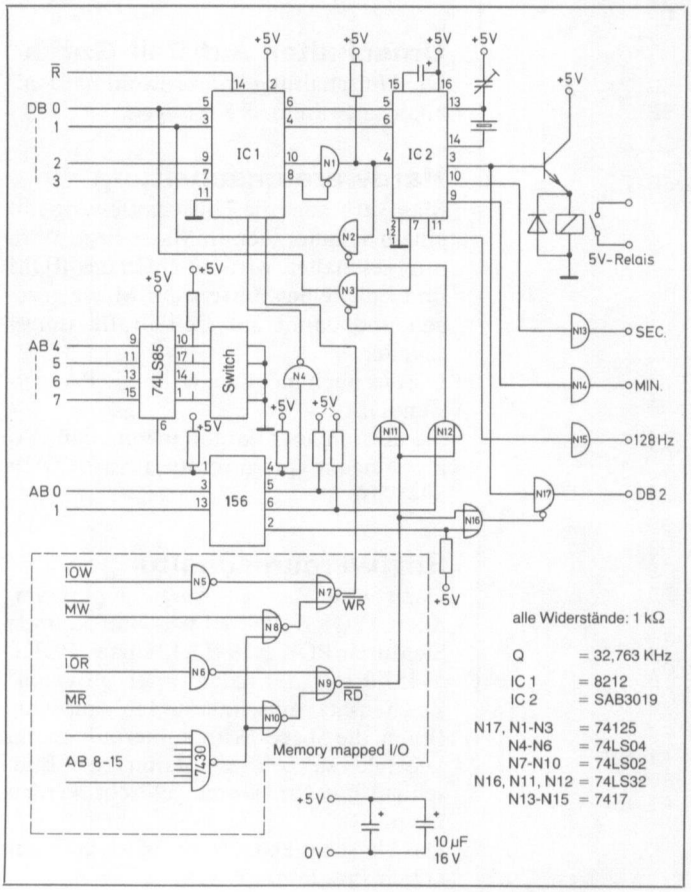
- serielle bidirektionale CBUS-Schnittstelle zur Ein-/Ausgabe von Minute, Stunde, Tag und Monat;
- Vorgeben einer Schaltzeit in ein Zeitregister, das bei Übereinkunft von Uhr/Datum oder nur der Uhrzeit mit dem Zeitregister einen Minutenimpuls zum Schalten eines externen Schaltelements gibt;
- zusätzliche Impulsausgänge für 128 Hz, 1 Hz und $^1/_{60}$ Hz im Taktverhältnis von 0,5;
- Pufferbetrieb mit Akku oder Batterie möglich (Kontrollausgang gibt an, ob Batteriespannung noch ausreicht);
- wahlweise Quarzoszillator oder Netzsynchronisation.

## Allgemeine Beschreibung

Kernstück der Schaltung ist der Uhren/Timer/Kalender-Baustein SAB3019. Der Rest der Schaltung ist wie eine normale Port-Schnittstelle ausgelegt und simuliert die VALVO eigene CBUS-Schnittstelle. Somit kann die Uhr/Timer/Kalender wie eine periphere Schaltung angesprochen werden.
Das Uhren-IC enthält eine Digitaluhr für Minute, Stunde, Tag und Monat sowie ein zusätzliches Register zur Vorgabe einer Schaltzeit ebenfalls für Minute, Stunde, Tag und Monat. Im Zeitzähler wird automatisch je nach Monatslänge ein Zyklus von 28, 30 oder 31 Tagen berücksichtigt. Durch einen Setzbefehl kann ein Schaltjahr berücksichtigt werden. Das IC ist für die Ansteuerung eines Mikrocomputers vorgesehen, und das Stellen und Auslesen der Register erfolgt über eine 3-Draht-CBUS-Schnittstelle. Der Mikrocomputer übernimmt die Aufbereitung der Daten und die Ansteuerung einer, wenn vorgesehen, Anzeigeeinheit.
Das Zeitregister ist ein 24-bit-Speicher, der ebenfalls über die CBUS-Schnittstelle gelesen oder gesetzt werden kann. Bei Übereinstimmung der Zeiten im Zeitzähler und im Zeitregister wird der Ausgang COMP für die Dauer einer Minute zu H-Pegel. Der Vergleich kann wahlweise mit oder ohne Datum stattfinden. Dies wird beim Setzen des Monatsregisters voreingestellt oder durch das Bit „NODA" gesteuert.

Vergleich mit Datum: UC = 0 **und** NODA = L
Vergleich ohne Datum: UC = 1 **oder** NODA = H

Der Datenverkehr von der Schaltung SAB3019 zum Mikrocomputer (Auslesen) und umgekehrt (Setzen) erfolgt über die Leitung „DATA" (Ein-/Ausgang), DLEN (Eingang) und CLB (Eingang). Daten und Adressen werden seriell über die Leitung DATA übertragen, gleichzeitig müssen Synchronisationsimpulse vom Mikrocomputer an CLB erzeugt werden. Die Länge der Datenübertragung wird durch die Zahl der CLB-Impulse, innerhalb des DLEN = H gegebenen Fenstern, bestimmt. Die Schaltung verfügt über eine interne Wortformat-




**Schaltplan für die Realisierung einer Uhr-, Timer- oder/und Kalenderfunktion**

```
Software:

Beispiele für den Programmablauf. Alle unterstrichenen Zeichen
müssen über die Console eingegeben werden.
Nach dem Programmstart erwartet das Programm die Eingabe eines
Charakters, das das entsprechende Unterprogramm auswählt.

1. Zeitregister lesen:
        T       TIME:  12:23
2. Datums- und Zeitregister lesen:
        D       DATE: o2:12      TIME:  12:23
3. Datums- und Zeitregister setzen:
        S       DATE: o2:12=o5:11 'CR'      TIME: 12:23=12:24 'CR'
   Wenn bei der Eingabe statt eines Zeichens ein 'DEL' engegben
   wird, bleibt der alte Inhalt des Registers erhalten.
4. Vergleichsregister ohne Datum setzen:
        R  CMP.DATE: N   DATE:  oo:oo=oo:oo
                         TIME:  oo:oo=12:12
5. Vergleichregister mit Datum setzen:
        R  CMP.DATE: Y   DATE:  oo:oo=o1:12
                         TIME:  12:12=12:13

In diesem Beispiel würde der Ausgang COMP am o1.12 um 12.13 Uhr
für eine Minute high.
```

Überprüfung: Es werden nur die nachfolgend aufgeführten Wortlängen als gültige Übertragung akzeptiert. Auf diese Weise lassen sich die Leitungen CLB, DATA und DLEN auch zum Ansteuern anderer Schaltungen verwenden.
Die Datenformate sind:

1. Adressieren ( 3 bit + Startbit)
2. Setzen (10 bit + Startbit)

Auf jeden Befehl „Adressieren" muß unmittelbar das Auslesen der Daten (7 bit) folgen. Der Befehl „Setzen" beinhaltet das Adressieren. Mit jedem Befehl (bzw. jedem Befehlszyklus Adressieren/Auslesen) lassen sich zwei Ziffern des Zeitzählers bzw. des Zeitregisters lesen oder setzen. Für ein vollständiges Auslesen bzw. Setzen sind somit vier Zyklen Adressieren/Auslesen oder vier Befehle Setzen notwendig.

## Das Auslesen:

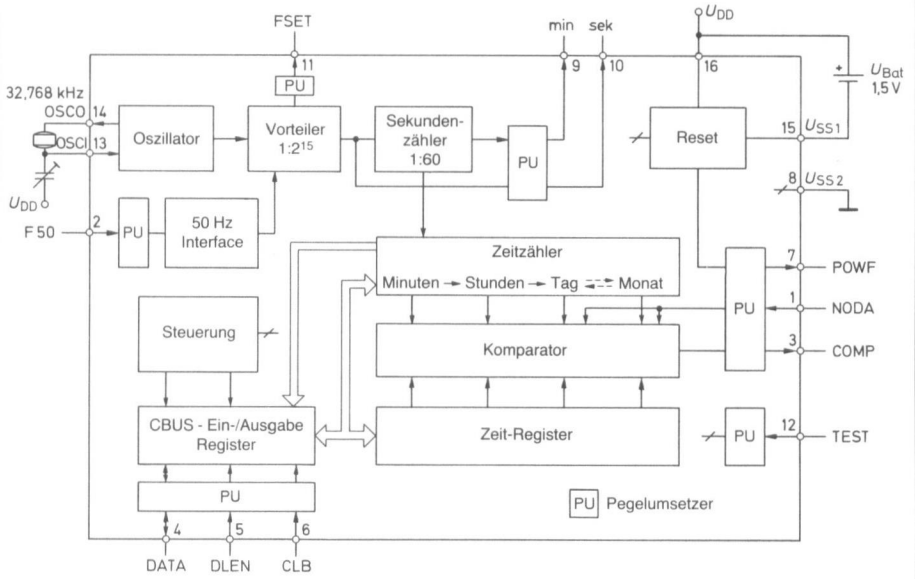
Zunächst werden mit dem Befehl Adressieren die Bit S, A0 und A1 vom Mikrocomputer zum SAB3019 übertragen, mit dem nächsten Befehl Auslesen wird dann der Inhalt der selektierten Dekaden vom SAB3019 zum Mikrocomputer im BCD-Kode übertragen.

## Das Setzen:

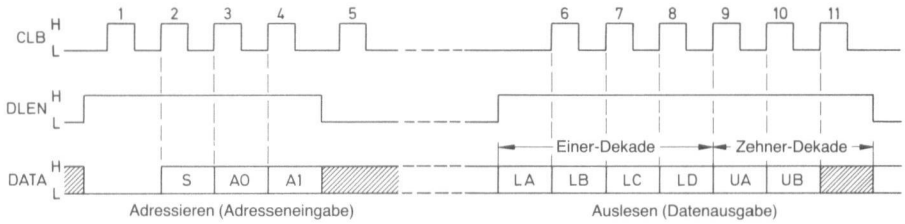
Mit dem Befehl Setzen werden sowohl die Adressen S,A0 und A1 als auch die in selektierten Dekaden einzuschreibenden Daten im BCD-Kode vom Mikrocomputer zum SAB3019 übertragen. Mit dem letzten Bit UC lassen sich hierbei Sonderfunktionen steuern.
Ein Setzbefehl für den Zeitzähler hält diesen nicht an, es wird auch kein Übertrag an eine nicht selektierte Einheit erzeugt.
Der Setzbefehl mit S = 0, A1 = 0 und A0 = 0 (Monat) und UC = 0 setzt Vorteiler und Sekundenzähler zurück. Falls der Sekundenzähler zwischen 30 und 59 steht, wird ein Übertrag für den Minutenzähler erzeugt. Mit diesem Befehl läßt sich die Uhr auf einfache Weise nachstellen, sofern die Abweichung kleiner als ±30 Sekunden ist.

## Die Schaltung:

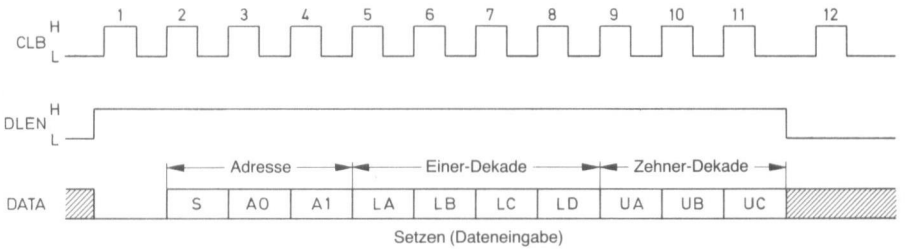
Die Gatter N5 – N10 und das IC7430 bilden die „Memory mapped I/0"-Einheit, wie sie bei 8080-Systemen üblich ist. Auch bei anderen Prozessorsystemen kann die Schaltung leicht funktionsfähig gemacht werden.
Die Bausteine 7485 und 74156 bilden die Port- oder Kanalnummer bzw. bilden drei benötigte $\overline{CE}$-Leitungen, eine $\overline{CE}$-




**Monolithisch integrierte CMOS-Schaltung SAB3019 von VALVO Uhr/Timer/Kalender für Mikrocomputer gesteuerte Systeme**




**Datenformat für einen Zyklus „Adressieren/Auslesen"**




**Datenformat für einen Befehl „Setzen"**

### Zuordnung der selektierten Dekaden zu den Adressenbit und Art der Auslesebefehle

| S | A0 | A1 | LA | LB | LC | LD | UA | UB | UC | Befehl | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | D | D | D | D | D | D | D | | Minute |
| 0 | 0 | 1 | D | D | D | D | D | D | 0/1 | Aus- | Stunde |
| 0 | 1 | 0 | D | D | D | D | D | D | 0/1 | lesen | Tag |
| 0 | 0 | 0 | D | D | D | D | D | 0/1 | 0/1 | Zähler | Monat |
| 1 | 1 | 1 | D | D | D | D | D | D | D | | Minute |
| 1 | 0 | 1 | D | D | D | D | D | D | 0/1 | Aus- | Stunde |
| 1 | 1 | 0 | D | D | D | D | D | D | 0/1 | lesen | Tag |
| 1 | 0 | 0 | D | D | D | D | D | 0/1 | 0/1 | Register | Monat |

```
Programm: U H R

Adresse  Obj.Code  Label   Statement  Comment
-------------------------------------------------------------------
  oAoo     97       START:  SUB A      ;Akkumulator loeschen
  oAo1     3231oC           STA HILF   ;HILF ist benötigter RAM-Speicher
  oAo4     DBCD             IN PORT 1  ;PORT 1, Schnittstelle Uhr
  oAo6     E6o1             ANI o1     ;Bit Test Power Failure
  oAo8     C2D7oA           JNZ POWF   ;gesetzt? Print ERROR MESSAGE
  oAoB     21B7oA           LXI H,TAB  ;HL-Reg auf TAB.Adresse setzen
  oAoE     CD1oo9           CALL TI    ;Eingabe eines Characters
  oA11     FE52             CPI 'R'    ;Ausgabe Register-Inhalt
  oA13     CA8BoB           JZ REGIS
  oA16     FE54             CPI 'T'    ;Ausgabe nur Zeit
  oA18     CA3DoA           JZ TIME
  oA1B     FE53             CPI 'S'    ;Setzen Datum und Zeit
  oA1D     CAF6oA           JZ SETZ
  oA2o     FE44             CPI 'D'    ;Ausgabe Zeit und Datum
  oA22     Co               RNZ        ;Rücksprung bei falscher Eingabe
  oA23     11E8oA   DATE:   LXI D,'DATE'
  oA26     CD2CoA           CALL S1o   ;Ausgabe des Datums
  oA29     11EFoA   So5:    LXI D,'TIME'
  oA2C     CDC8oB   S1o:    CALL TAB   ;Tabulator Ausgabe
  oA2F     CDBBoB   S15:    CALL TEXT  ;Textausgabe
  oA32     CD43oA   S2o:    CALL AUSL  ;gewähltes Register auslesen
  oA35     oE3A             MVI C,':'  ;Doppelpunkt ausgeben
  oA37     CD9oo5           CALL CO
  oA3A     C343oA           JMP AUSL   ;
  oA3D     21C7oA   TIME:   LXI H,TAB+16 ;Zeit ausgeben
  oA4o     C329oA           JMP So5    ;
  oA43     o6o6     AUSL:   MVI B,o6   ;Zähler für die ersten sechs
  oA45     CD88oA           CALL A1o   ;Zeichenausgaben setzen
  oA48     o6o6             MVI B,o6
  oA4A     16oo             MVI D,oo   ;D-Reg. löschen
  oA4C     DBCE     BET:    IN PORT1   ;Unterprogramm zum Einlesen von
  oA4E     E6o2             ANI o2     ;sieben Zeichen
  oA5o     82               ADD D      ;
  oA51     o7               RLC        ;
  oA52     57               MOV D,A    ;
  oA53     3EoB             MVI A,oB   ;
  oA55     D3CF             OUT PORT1  ;
  oA57     3Eo9             MVI A,o9   ;
  oA59     D3CF             OUT PORT1  ;
  oA5B     o5               DCR B      ;
  oA5C     C24CoA           JNZ BET    ;alle Zeichen gelesen? Nein,jump!
  oA5F     DBCE             IN PORT1   ;letztes Zeichen lesen!
  oA61     E6o2             ANI o2     ;
  oA63     82               ADD D      ;
  oA64     57               MOV D,A    ;Akkuinhalt retten
  oA65     97               SUB A      ;Akkumulator löschen
  oA66     D3CF             OUT PORT1  ;Schnittstelle schließen
  oA68     o6o3             MVI B,o3   ;
  oA6A     CD72oA           CALL ZEICH;Unterprogramm zur seriellen Daten-
  oA6D     o6o4             MVI B,o4   ;erfassung
  oA6F     C375oA           JMP ZEI1o  ;
  oA72     7A       ZEICH:  MOV A,D    ;
  oA73     oF               RRC        ;
  oA74     57               MOV D,A    ;
  oA75     7A       ZEI1o:  MOV A,D    ;
  oA76     oF               RRC        ;
  oA77     57               MOV D,A    ; Akkuinhalt retten
  oA78     7B               MOV A,E    ;
  oA79     17               RAL        ;
  oA7A     5F               MOV E,A    ;Akkuinhalt retten
  oA7B     o5               DCR B      ;
  oA7C     C275oA           JNZ ZEI1o  ;
  oA7F     7B               MOV A,E    ;
  oA8o     A6               ANA M      ;
  oA81     23               INX H      ;
  oA82     C63o             ADI 3o     ;Zahl in ASCII-Code bringen
  oA84     4F               MOV C,A    ;Zeichen zur Ausgabe ins C-Reg.
  oA85     C39oo5           JMP CO     ;Zeichen ausgeben
  oA88     3Eo1     A1o:    MVI A,o1   ;
  oA8A     D3CF             OUT PORT1  ;
  oA8C     48               MOV C,B    ;
  oA8D     oD               DCR C      ;
  oA8E     56       A12:    MOV D,M    ;
  oA8F     78               MOV A,B    ;
  oA9o     B9               CMP C      ;
  oA91     C29FoA           JNZ A15    ;
  oA94     3A31oC           LDA HILF   ;
  oA97     FEoo             CPI oo     ;
  oA99     CA9FoA           JZ A15:    ;
  oA9C     3E75             MVI A,75   ;
  oA9E     57               MOV D,A    ;
  oA9F     CDA8oA   A15:    CALL A2o   ;
  oAAo     23               INX H      ;
  oAA1     o5               DCR B      ;
  oAA2     C28EoA           JNZ A12    ;
  oAA5     C9               RET        ;
  oAA8     7A       A2o:    MOV A,D    ;
  oAA9     oF               RRC        ;
  oAAA     oF               RRC        ;
  oAAB     oF               RRC        ;
  oAAC     oF               RRC        ;
  oAAD     E6oF             ANI oF     ;
  oAAF     D3CF             OUT PORT1  ;
  oAB1     7A               MOV A,D    ;
  oAB2     E6oF             ANI oF     ;
  oAB4     D3CF             OUT PORT1  ;
  oAB6     C9               RET        ;
  oAB7     31317531 TAB:    DB 31H,31H,75H,31H,o2H,o9H,o3H,oFH
  oABB     o2o9o3oF
  oABF     31313131         DB 31H,31H,31H,31H,o2H,o9H,o1H,oFH
  oAC3     o2o9o1oF
  oAC7     31313175         DB 31H,31H,31H,75H,o2H,o9H,o3H,oFH
  oACB     o2o9o3oF
  oACF     31317575         DB 31H,31H,75H,75H,o2H,o9H,o7H,oFH
  oAD3     o2o9o7oF
  oAD7     11DDoA   POWF:   LXI D,'POWT';Ausgabe der Fehlermeldung
  oADA     C3BBoB           JMP TEXT
  oADD     5o4F5746 POWT:   .ASCII /Power Failure High/
  oAE1     2o484947
  oAE5     482oA1
  oAE8     44415445 DATE:   .ASCII /DATE:  /
```

### Zuordnung der selektierten Dekaden zu den Adressenbit und Art der möglichen Setzbefehle

| S | A0 | A1 | LA | LB | LC | LD | UA | UB | UC | Befehl |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | D | D | D | D | D | D | D | Setzen Minutenzähler |
| 0 | 0 | 1 | D | D | D | D | D | D | X | Setzen Stundenzähler |
| 0 | 1 | 0 | D | D | D | D | D | D | X | Setzen Tageszähler |
| 0 | 0 | 0 | D | D | D | D | D | X | 1 | Setzen Monatszähler |
| 0 | 0 | 0 | X | X | X | X | X | X | 0 | Sek.zähler reset |
| 1 | 1 | 1 | D | D | D | D | D | D | D | Setzen Min. Register |
| 1 | 0 | 1 | D | D | D | D | D | D | X | Setzen Std.-Register |
| 1 | 1 | 0 | D | D | D | D | D | D | X | Setzen Tag.-Register |
| 1 | 0 | 0 | D | D | D | D | D | X | 0 | Setzen Monatsregister Vergleich **mit** Datum |
| 1 | 0 | 0 | D | D | D | D | D | X | 1 | Setzen Monatsregister Vergleich **ohne** Datum |

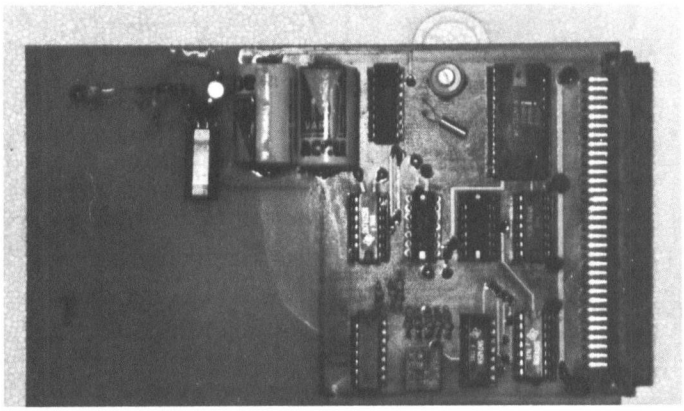
Leitung zum Setzen der Uhrenbausteine, zwei zum Lesen.

Die Übertragung der seriellen Bit geschieht über das IC8212 von Intel, was einem 8fach-Latch entspricht, es können also einzelne Bit gesetzt bzw. rückgesetzt werden. Hierbei bildet das Bit 0 des Datenwortes das Signal DLEN des Uhrenbausteines, Bit 1 entspricht CLB, und Bit 2 steht für DATA. Das Gatter N1 muß beim Auslesen des Uhren-IC den Tri-State für den Datenbus herstellen, das heißt, wird das IC2 gesetzt, muß Bit 3 des Datenwortes L-Pegel sein, wird das Uhren-IC ausgelesen, muß dasselbe Bit einen H-Pegel annehmen, also disabled werden.

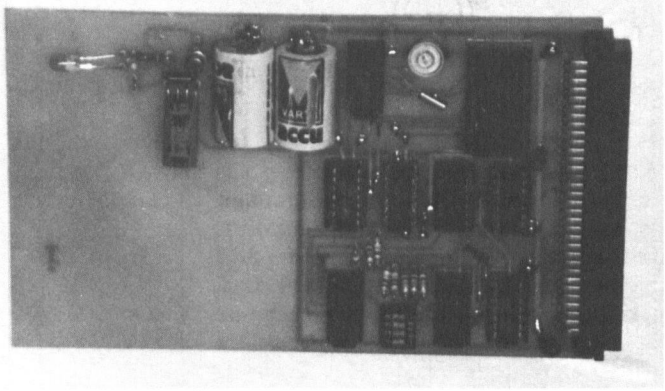
Mit einem Read-Impuls und dem Aktivieren von Gatter N2 können die Daten auf den Datenbus des Prozessorsystems gebracht werden. Über das Gatter N3 kann der Zustand des Uhren-IC abgefragt werden. Bei Gleichheit des Zeitzählers und des Zeitregisters wird der Ausgang an Pin 3 zu H-Pegel, schaltet den Transistor T1 durch, das Relais zieht an, und es können irgendwelche Funktionen erzeugt werden.

```
dresse  Obj.Code  Label  Statement    Comment

AEC     3A2oAo
AEF     54494D45 TIME:    .ASCII /TIME:  /
AF3     3A2oAo
AF6     97         SETZ:  SUB A          ;Akkumulator loeschen
AF7     3231oC            STA HILF      ;Hilfsspeichen loeschen
AFA     11E8oA     SET:   LXI D,'DATE'  ;
AFD     CD2CoA            CALL S1o      ;Ausgabe Text 'DATE'
Boo     oE3D              MVI C,':'     ;Doppelpunkt ausgeben
Bo2     CD9oo5            CALL CO       ;Consol-Output Subroutine
Bo5     E5                PUSH H        ;HL-Reg retten
Bo6     CD1EoB            CALL REGDI    ;Zeichen einlesen
Bo9     E1                POP H         ;
BoA     CD29oA            CALL So5      ;Timer-Inhalt ausgeben
BoD     oE3D              MVI C,':'     ;Doppelpunkt ausgeben
BoF     CD9oo5            CALL CO       ;Consol-Output Subroutine
B12     21C7oA            LXI H,TAB+16  ;HL-Reg. mit TAB laden
B15     CD66oB     SET1o: CALL ZEITN    ;Zeichen über Konsole eingeben
B18     FE4F              CPI 'DEL'     ;Break-Charakter?
B1A     C8                RZ            ;Return, wenn ja
B1B     C333oB            JMP D1o       ;Zeichen an Uhr ausgeben
B1E     21B7oA     REGDI: LXI H,TAB     ;HL-Reg. mit TAB laden
B21     CD66oB     Do5:   CALL ZEITN    ;Zeichen über Konsole eingeben
B24     FE4F              CPI 'DEL'     ;Break-Character?
B26     C8                RZ            ;Return, wenn ja
B27     3A31oC            LDA HILF      ;Test Bit 1 in Hilf
B2A     FEo2              CPI o2        ;
B2C     CA33oB            JZ D1o        ;
B2F     7A                MOV A,D       ;Bit 7 in Reg. D setzen
B3o     F64o              ORI 4o        ;
B32     57                MOV D,A       ;
B33     o6o4       D1o:   MVI B,o4      ;
B35     D5                PUSH D        ;DE-Reg. retten
B36     CD88oA            CALL A1o      ;Zeichen an Uhr ausgeben
B39     7B                MOV A,E       ;
B3A     CD47oB            CALL AUS      ;
B3D     11o4oo            LXI D,ooo4    ;Lade DE-Reg. mit 4
B4o     19                DAD D         ;HL-Reg. um 4 erhöhen
B41     o6o4              MVI B,o4      ;
B43     CD88oA            CALL A1o      ;
B46     F1                POP PSW       ;Inhalt D-Reg. in Akkumulator
B47     o6o7       AUS:   MVI B,o7      ;
B49     oF         AUSo5: RRC           ;
B4A     F5                PUSH PSW      ;
B4B     DA53oB            JC AUS1o      ;
B4E     3E31              MVI A,31      ;
B5o     C355oB            JMP AUS2o     ;
B53     3E75       AUS1o: MVI A,75      ;
B55     57         AUS2o: MOV D,A        ;
B56     CDA8oA            CALL A2o      ;
B59     F1                POP PSW       ;
B5A     o5                DCR B         ;
B5B     C249oB            JNZ AUSo5     ;
B5E     97                SUB A         ;Akkumulator loeschen
B5F     D3CF              OUT PORT1     ;Schnittstelle schließen
B61     162o              MVI D,2o      ;
B63     C3A8oA            JMP A2o       ;
```

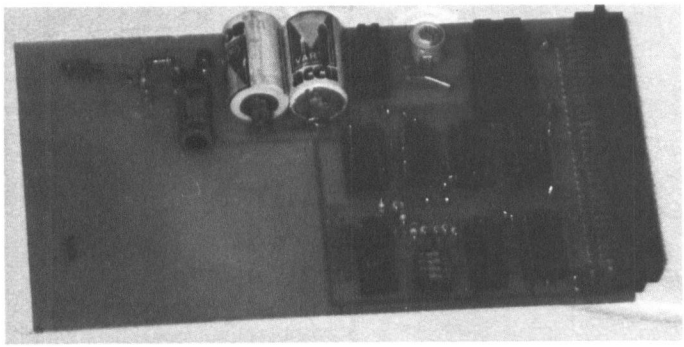
```
Adresse  Obj.Code  Label  Statement    Comment

oB66     16oo      ZEITN: MVI D,oo     ;
oB68     CD1oo9    Z1o:   CALL TI      ;Consol-Input Subroutine
oB6B     D63o             SUI 3o       ;Ascii in Binär wandeln
oB6D     FEoA             CPI ':'      ;Doppelpunkt ?
oB6F     CA87oB           JZ WEX       ;D-Reg. gegen E-Reg. tauschen
oB72     FEDD             CPI 'CR'     ;Vergleich Carriage Return
oB74     C8               RZ           ;
oB75     FE4F             CPI 'DEL'    ;Break-Character?
oB77     C8               RZ           ;Return, wenn ja
oB78     F5               PUSH PSW     ;Akku-Inhalt retten
oB79     7A               MOV A,D      ;D-Reg. mit 16 multiplizieren
oB7A     o7               RAL          ;
oB7B     o7               RAL          ;
oB7C     o7               RAL          ;
oB7D     o7               RAL          ;
oB7E     E6Fo             ANI Fo       ;Bit o bis 3 loeschen
oB8o     57               MOV D,A      ;
oB81     F1               POP PSW      ;
oB82     82               ADD D        ;
oB83     57               MOV D,A      ;
oB84     C368oB           JMP Z1o      ;Ende Zeicheneingabe
oB87     5A         WEX:  MOV E,D      ;erstes Zeichen retten
oB88     C366oB           JMP ZEITN    ;nächstes Zeichen eingeben
oB8B     3Eo1       REGIS: MVI A,o1    ;Vergleichsregister setzen
oB8D     3231oC           STA HILF     ;
oB9o     11A7oB           LXI D,'CMP'  ;Textausgabe
oB93     CDBBoB           CALL TEXT    ;
oB96     CD1oo9           CALL TI      ;Consol-Input Subroutine
oB99     FE59             CPI 'Y'      ;Vergleich mit Datum
oB9B     CAB3oB           JMP YES      ;
oB9E     FE4E             CPI 'N'      ;Vergleich ohne Datum
oBAo     Co               RNZ          ;Return bei falscher Eingabe
oBA1     CD6Do6           CALL CRLF    ;Ausgabe 'CR' und 'LF'
oBA4     C3FAoA           JMP SET      ;
oBA7     2o2o434D CMP:    .ASCII /CMP.DATE: /
oBAB     5o2E4441
oBAF     54453AAo
oBB3     3Eo2       YES:  MVI A,o2     ;Hilf setzen für Datum
oBB5     3231oC           STA HILF     ;
oBB8     C3FAoA           JMP SET      ;

oBBB     1A         TEXT: LDAX D       ;ersten Zeichen vom Text laden
oBBC     4F               MOV C,A      ;
oBBD     CD9oo5           CALL CO      ;Consol-Output Subroutine
oBCo     79               MOV A,C      ;
oBC1     E68o             ANI 8o       ;Bit o bis 6 ausblenden
oBC3     Co               RNZ          ;Return, wenn Bit 7 gesetzt
oBC4     13               INX D        ;nächstes Zeichen holen
oBC5     C3BBoB           JMP TEXT     ;nächstes Zeichen ausgeben

oBC8     C5         TAB:  PUSH B       ;Reg. BC retten
oBC9     o6o5       TAB1o: MVI B,o5    ;
oBCB     oE2o       TAB2o: MVI C,' '   ;Lade C-Reg. mit Blank
oBCD     CD9oo5           CALL CO      ;Consol-Output Subroutine
oBDo     o5               DCR B        ;
oBD1     C8               RZ           ;
oBD2     C3CBoB           JMP TAB2o    ;
```



**Gesamtansicht der Platine für Uhr/Timer/Kalender**



**Anordnung der integrierten Bausteine**



**CMOS-Baustein 3019 mit Puffer-Batterien**

Die Taktausgänge (Pin 9, 10, 11) können über Treiberschaltungen zu anderen Test- oder Steuerzwecken herangezogen werden.

Der Nachbau dieser Schaltung dürfte keinerlei Schwierigkeiten aufweisen, da sie sehr einfach ist.

Beim Erstellen der nötigen Software müssen die angegebenen Intervalle von Setzen und Lesen beachtet werden, ansonsten dürften auch hier keine Probleme auftreten.

Bernhard Strobl und Robert Valentan

# Computeruhr

Um einem Computer das Gefühl der Zeit beizubringen, gibt es im Prinzip zwei Möglichkeiten. Erstens eine Softwarelösung, die ein ziemlich umfangreiches und kompliziertes Programm benötigt. Ein Zähler (er muß sich auf der Computerplatine befinden, z.B. 16 bit im 6522) wird so eingestellt, daß er in gewissen Zeitabständen einen Interrupt auslöst. Dieser Interrupt wird nun folgendermaßen behandelt. Bei jedem Interrupt wird die Uhr (eine Variable) ein Stück weitergesetzt (aufsummiert), eben um so viel, wie die Zeitabstände der Interrupts betragen. Diese Zeit ist nun proportional zur gewünschten Zeit und muß „nur noch" in Stunden, Minuten und Sekunden unterteilt werden. Diese Lösung hat zwei wesentliche Nachteile. Wird der Computer ausgeschaltet, so bleibt die Zeit nicht vorhanden, und außerdem muß das Programm neu geladen werden. Versucht man dies zu umgehen und speichert das Programm fix in einen EPROM (PROM), so bleibt das Problem, daß die Uhr nach dem Einschalten neu synchronisiert werden muß. Auch die Genauigkeit läßt zu wünschen übrig, da nach dem Einschalten der Quarz noch kalt ist und seine eigentliche Frequenz noch nicht ganz erreicht hat. Auch dadurch ergeben sich Ungenauigkeiten. Handelt es sich noch dazu um zeitkritische Programme, die nebenbei laufen, bei denen es auf jeden Taktzyklus ankommt, fällt diese Lösung sowieso aus. Ganz anders ist eine Hardware-Lösung. Hier bleibt die Zeit ähnlich einer normalen Digitaluhr erhalten, die einzelnen Digits werden einfach nacheinander abgefragt. Auch beim Ein- und Ausschalten ergeben sich keine Probleme, wenn die Schaltung über eine Batterie betrieben wird. Der Übertrag bei 60 Sekunden (Minuten) ist hardwaremäßig gelöst. So reifte in uns die Idee, eine solche Uhr zu bauen. Sie sollte folgende Voraussetzungen erfüllen:

- einfacher Anschluß über Adreßbus und Datenbus (nicht über Portbaustein)
- Pufferbetrieb (5 Volt, geringe Stromaufnahme)
- nicht mehr als acht Adressen belegt
- im Adressenbereich frei verschiebbar von 0000 bis FFFF
- Funktionen: einfaches Einlesen der Zeit (Digits) in die Register, Start/Stopp/Setzen/Rücksetzen auf 00 00 00

## Der Taktgenerator

Diese Aufgabe erfüllt ein einziger IC. Es ist der 8650 von Seikosha. In ihm befindet sich ein Quarzoszillator, der eine stabile Frequenz erzeugt. Durch geeignetes Beschalten der Kontrolleingänge kann die Frequenz am Ausgang verändert werden. Um exakt 1 Hertz am Ausgang zu erhalten, müssen die Anschlüsse 3, 4 und 7 auf + gelegt werden. Auf jeden Fall muß die Resetleitung (Pin 14) auf + liegen.

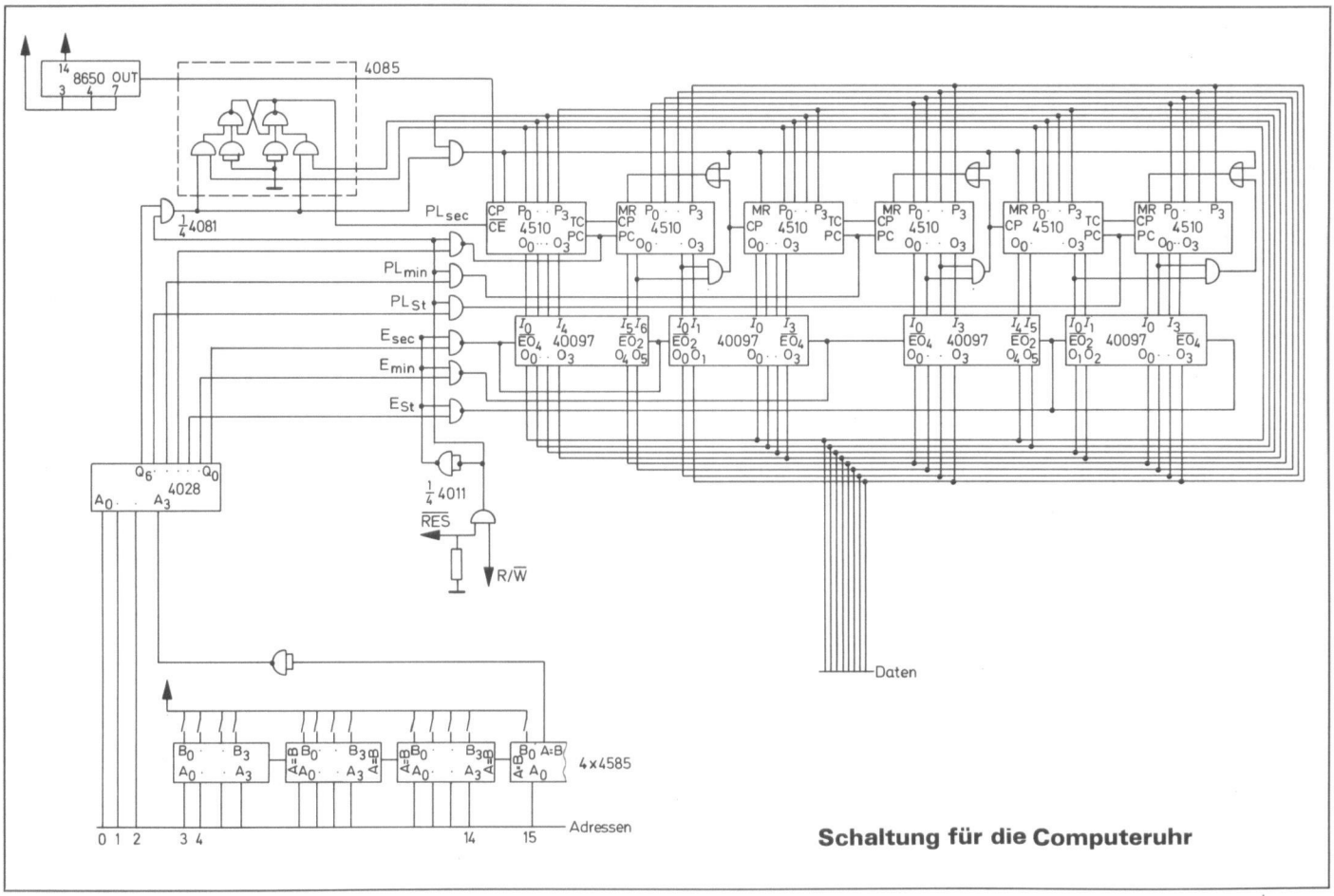
```
(k)*=o3oo
/38
 o3oo ao ldy #oo
 o3o2 a2 ldx #oo
 o3o4 e8 inx
 o3o5 fo beqo31o
 o3o7 oe asl ffff
 o3oa oe asl ffff
 o3od 4c jmp o3o4
 o31o 2o jsr o31a
 o313 c8 iny
 o314 fo beq o319
 o316 4c jmp o3o2
 o319 oo brk
 o31a 2o jsr eb44
 o31d 2o jsr e83b
 o32o ad lda 8oo2
 o323 2o jsr o339
 o326 a9 lda #3a
 o328 2o jsr efo5
 o32b ad lda 8oo1
 o32e 2o jsr o339
 o331 a9 lda #3a
 o333 2o jsr efo5
 o336 ad lda 8ooo
 o339 8d sta offf
 o33c 29 and #fo
 o33e 4a lsr .a
 o33f 4a lsr .a
 o34o 4a lsr .a
 o341 4a lsr .a
 o342 18 clc
 o343 69 adc #3o
 o345 2o jsr efo5
 o348 ad lda offf
 o34b 29 and #of
 o34d 18 clc
 o34e 69 adc #3o
 o35o 2o jsr efo5
 o353 6o rts
```

## Die eigentliche Uhr

Um die Uhr auch auf jede Zeit setzen zu können, wurden programmierbare Zähler verwendet. Der 4510 ist ein 4-bit-Zähler mit einem Parallel-Laden-Eingang. Setzt man diesen auf H-Pegel, so werden die Eingangszustände der Eingänge $P_0$ bis $P_3$ in den Zähler übernommen. Außerdem besitzt er einen Count-Enable-Eingang (CE), eine Master-Reset-Leitung (MR), eine Übertragungsleitung bei 10 und eine Auf- und Abzählsteuerung. Letztere wird bei unserer Uhr allerdings nicht verwendet. Für eine 24-Stunden-Uhr benötigt man sechs Zähler. Zwei davon bilden jeweils den Sekunden-, den Minuten- und den Stundenzähler. Der Minutenzähler darf nur alle 60 Sekunden einen Taktimpuls bekommen, gleichzeitig muß der Sekundenzähler auf 00 gesetzt werden. Um das zu bewerkstelligen, werden das zweite und das dritte Bit des Sekundenzählers einem UND-Gatter zugeführt, dessen Ausgang nur dann kurzzeitig auf H-Pegel überspringt, wenn der Ausgangszustand des Sekundenzählers XXXX XIIX ergibt. Da die Uhr von unten heraufzählt, werden die Sekunden automatisch bei 60 auf 00 zurückgesetzt, da der MR-Eingang des Zählers mit dem Ausgang des UND-Gatters verbunden ist. Gleichzeitig bekommt auch der Minutenzähler einen Impuls. Das gleiche passiert beim Minutenzähler. Beim Stundenzähler muß die Bitkombination anders aussehen, um ein Rücksetzen der Uhr bei 24 Uhr zu ermöglichen (XX1X X1XX). Beim TC-Ausgang des letzten Zählers können Tagesimpulse abgezapft werden (Erweiterung als Kalender). Um auch ein Rücksetzen der Uhr von außen zu erlauben, müssen die Rücksetzleitungen der Zählergruppen mit ODER-Gatter verknüpft werden. Einer- und Zehnerzähler des Sekundenzählers ergeben beispielsweise zusammen eine 8 bit breiten Datenbus, der dann dem Computer zugeführt wird.

## Adaptierung an den µP-Tri-State-Bus

Insgesamt müssen 3 × 8 bit dem µP übergeben (beim Abfragen der Zeit) und 3 ×



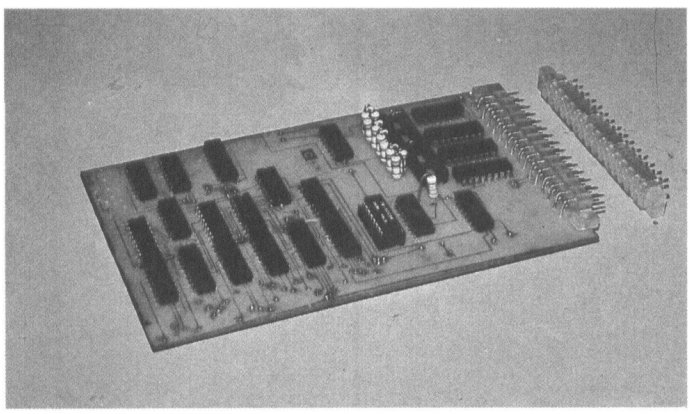

**Schaltung für die Computeruhr**

8 bit übernommen werden (beim Setzen der Uhr). Das Ausgeben der Daten erfolgt über Tri-State-Buffer des Bausteines 40097. Dieser Baustein besitzt noch zusätzlich eine Freigabeleitung, die es ermöglicht, die Daten am Eingang erst nach dem Freigabeimpuls am Ausgang erscheinen zu lassen. Ein Ausgeben der Daten von einem der drei Zähler wird erreicht, indem die Enable-Leitungen der richtigen Buffer aktiviert werden.

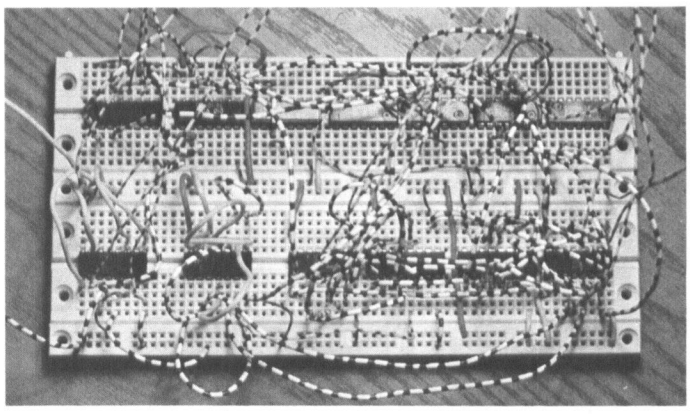
Das Setzen der Zeit funktioniert ähnlich. Hier sind die Freigabeschaltungen schon im Zähler integriert. Die 8 bit am Datenbus werden in den Zähler übernommen, dessen Parallel-Load-Eingang auf H liegt. Zusätzlich wurde auch die R/$\overline{W}$-Leitung des µP in die Uhr übernommen und mit UND- und NAND-Gattern verknüpft. So ist bei einer Datenübertragung des µP zur Uhr (write) ein Aktivieren der Buffer unmöglich, da es sonst zu einem Kurzschluß kommen kann.

## Die Adressendekodierung

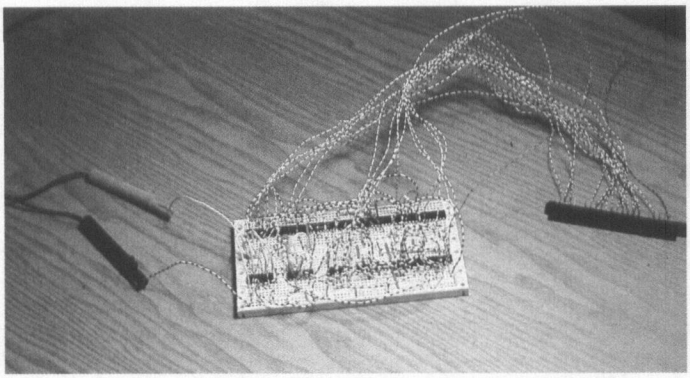
Da die Uhr im ganzen Adressenbereich frei verschieblich sein sollte, wurde diese Aufgabe mit vier 4-bit-Vergleichern gelöst. Der 4585 ist kaskadierbar und ermöglicht daher ein Vergleichen aller oberen 13 Adressen. Mit den verbleibenden unteren drei kann man acht Kombinationen erzeugen. Diese drei Adressen werden an einen BCD-zu-10-Dekoder angeschlossen. Je nachdem wie das Bitmuster dieser drei Adressen aussieht, führt einer der Ausgänge von $O_0$ bis $O_7$ einen H-Pegel. Bis auf einen Ausgang werden alle für die verschiedenen Befehle benötigt. Um ein Umadressieren der Uhr leicht zu verwirklichen, wurden die anderen Eingänge der Vergleicher an zwei IC-Sokkeln angeschlossen. Durch Einstecken



**Aufbau der Platine**



**Testaufbau der Computeruhr**



**Platinenaufbau mit Sockelleiste für die Computeruhr**

einer Drahtbrücke kann man leicht die Adresse ändern. Die Funktion des NAND-Gatters wird klar, wenn man den vierten Eingang des 4028 im L-Pegel hält, um die Ausgänge 0 – 7 anzusprechen. Ein L-Pegel an dem NAND-Gatter wird dann erreicht, wenn vor dem NAND ein H-Pegel liegt, der wiederum nur vorhanden ist, wenn die richtige Adresse ansteht.

## Start, Stopp und Reset

Ein Flipflop wird durch den Computer gesetzt und gelöscht. Der Ausgang des Flipflop wird mit dem Count-Enable-Eingang des ersten Zählers verbunden. Der siebente Ausgang des 4028 erfüllt drei Aufgaben: Starten und Stoppen der Uhr bei einem Masterreset-Befehl an alle Zähler. Da diese drei Befehle keinen Datentransfer benötigen, kann man den Datenbus zur weiteren Dekodierung verwenden. Ist das erste Bit auf 1-Signal, so startet die Uhr, ist das zweite Bit auf 1-Signal, so stoppt sie oder umgekehrt, je nachdem ob Q oder Q mit dem $\overline{CE}$-Eingang des ersten Zählers verbunden ist. Das Flipflop mußte so umständlich aufgebaut 4085 werden, da andere Flipflops zu langsam waren. Ist das dritte Bit auf 1-Signal, so bekommt jeder Zähler einen Reset-Befehl, und die Uhr wird zurückgestellt.

## Die Software

Ganz ohne Software kommt man natürlich nicht aus, wenn man die Zeit abfragen will. Ob man jetzt das BASIC mit PEEK und POKE verwendet oder ein Assemblerprogramm schreibt, kommt auf den Verwendungszweck an.

## Ein Beispiel

Wir plazierten unsere Uhr auf 8000H. Der Programmstart erfolgt mit der Funktionstaste F1, dem AIM65. Die Zeit wird kurz angezeigt und verschwindet dann wieder. Die Uhr zählt weiter. Das Ganze ist hier nur vom Monitor aus aufrufbar. Zum Programmieren, Starten und Stoppen können die Monitorroutinen des AIM zum Speicherabfragen/-ändern verwendet werden, da die Uhr wie ein normaler RAM angesprochen wird.

Thomas Franz

# Homecomputer mit EUROCOM II

Ziel des Projektes ist der Bau eines kompletten Homecomputers.
Der Selbstbau hat im wesentlichen zwei Vorzüge:

- eigene Spezifikationen,
- Lerneffekt.

Eine beträchtliche Ersparnis gegenüber kommerziellen Rechnern ergibt sich meistens nicht, jedoch kann eine höhere Leistungsfähigkeit bei gleichem Preis erzielt werden.
Hier betragen die Gesamtkosten etwa 2790 DM.
Die Selbstentwicklung eines ganzen Computers stößt meist schnell an – insbesondere technische – Grenzen, da:

- Platinen mit ausreichender Packungsdichte privat nicht herstellbar sind,
- geeignete Meßmittel nicht zur Verfügung stehen.

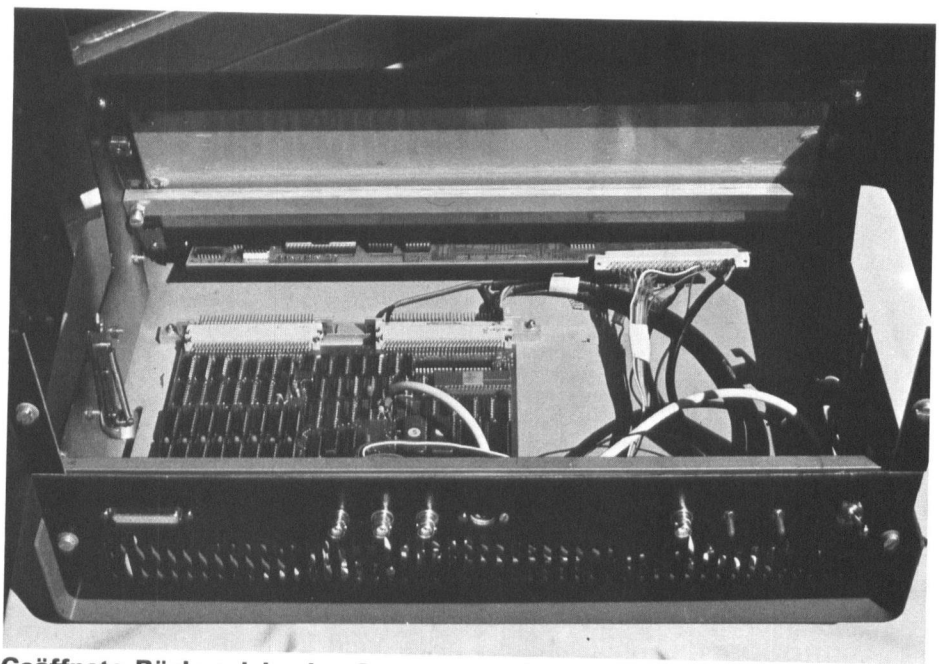
Man entschließt sich daher, sich nach Bausatz- oder Teilfertigsystemen umzusehen, deren Auswahl inzwischen groß ist.
Als CPU wählte ich den bekannten 6809 von Motorola.
Die wichtigsten Eigenschaften des MC6809:
8-bit-CPU mit 16-bit-Verarbeitung,
64K-Adressenraum,
59 Mnemonische Kodes,
268 Opkodes Operationkodes,
1464 Instruktionen mit verschiedenen Adressierungsarten.
Teilfertigsysteme mit MC6809 bietet der EUROCOM II der Fa. ELTEC. Hier kurz die Eigenschaften:
CPU: MC6809 1,33 MHz Takt,
48K-RAM enthält 16K-Grafik (256 × 512 Punkte),
32 Zeilen à 64 Zeichen darstellbar
RS-232-Schnittstelle (z.B. für Druckeranschluß),
Tastaturanschluß,
300-Bd-Kassettenrecorder-Anschluß,
BAS-Video-Komposite-Signal (Sync auch einzeln verfügbar),
ASCII-Zeichensatz (beliebige Zeichen definierbar),
Grafik-Software auf der Platine,
4 EPROM-Steckplätze, wahlweise mit 4K (2708) oder 8K (2716) oder 16K (2732) bestückbar,
in der neuesten Version mit 2 × 2K-Monitor belegt, so daß 4K-EPROM noch frei verfügbar.
Nun hat man zwar eine schöne Platine, aber noch längst keinen Computer. Dieser Artikel beschreibt **daher** den Bau einer „Hülle" für den EUROCOM II.

## Die Tastatur

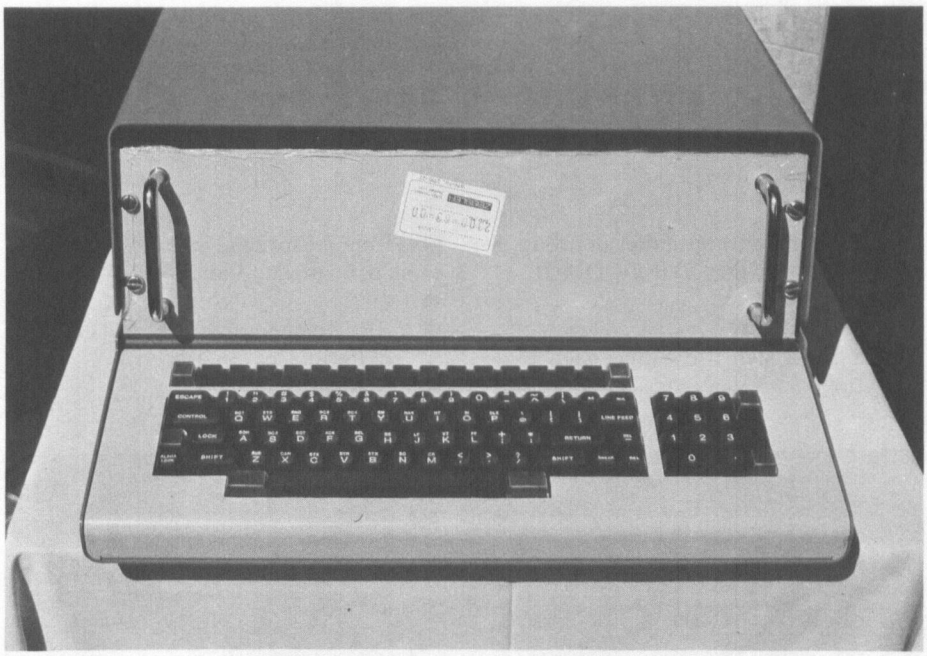
Zuerst stellt sich die Frage nach einer Tastatur.
Folgende Ansprüche soll diese haben:

- selbstbaufreundlich
- robust
- voller Zeichensatz (Umlaute evtl. mit frei definierbaren Tasten)
- einfach anschließbar

**Geöffneter Rechner mit dem selbstgebauten Netzteil**



**Geöffnete Rückansicht des Computers mit der Platine des EUROCOM II**



**Vorderansicht des Computers mit der Tastatur**

Am einfachsten ist es, nach einem Fertigprodukt zu greifen. Meine Wahl fiel auf: CHERRY CB80-12AA, die zwar nicht billig ist, dafür aber fast allen Ansprüchen gerecht wird:

- kapazitive Tasten
- 96-Tasten-ASCII-Zeichensatz (groß, klein, Spezialzeichen)
- Shiftlock, Alphalock
- abgesetzte Zehner-Tastatur
- Sonderfunktionstasten 23 (frei verwendbar)
- parallele Ausgabe mit Speicherung der zuletzt gedrückten Taste

## Das Gehäuse

Nach längerer Suche erwies sich das abgebildete ZEISSLER-Gehäuse als brauchbar: Die relativ große Tastatur findet Platz, und der Deckel ist ohne Werkzeug abnehmbar.

Allein das Netzteil bleibt dem vollständigen Selbstbau überlassen.

Anforderungen:

- 5 V/2 A, 12 V/1 A, – 12 V/0,5 A, – 5 V/0,5 A.
- Die – 5 V müssen zuerst ein- und zuletzt ausgeschaltet werden.

Das Netzteil hat folgende Eigenschaften:

- 5 V/5 A, 12 V, – 12 V, – 5 V je 1,5 A
- Überspannungsschutz

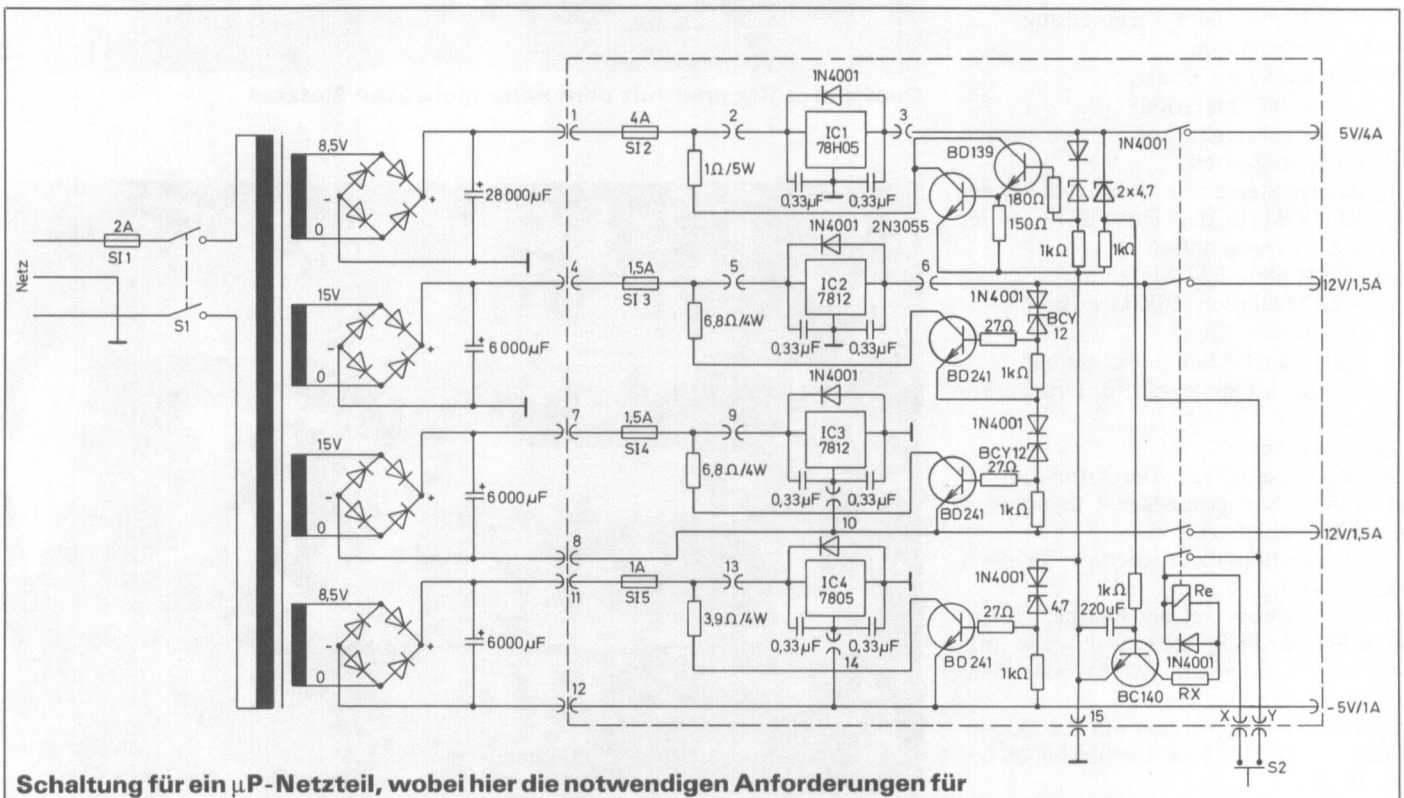
## Die Netzteilschaltung

Der Stromlaufplan zeigt zunächst die Stabilisierung, die nur mit Positiv-Reglern aufgebaut ist, da diese besser erhältlich sind; außerdem ergeben sich schaltungstechnische Vorteile.

Nach Gleichrichtung und Siebung werden die Spannungen durch die Regler stabilisiert. Diese sind mit Kondensatoren am Ein- und Ausgang entkoppelt. Die gegengeschaltete Diode verhindert einen rückfließenden Strom im Falle eines Kurzschlusses am Eingang, der u.U. zur Zerstörung des Reglers führen würde.

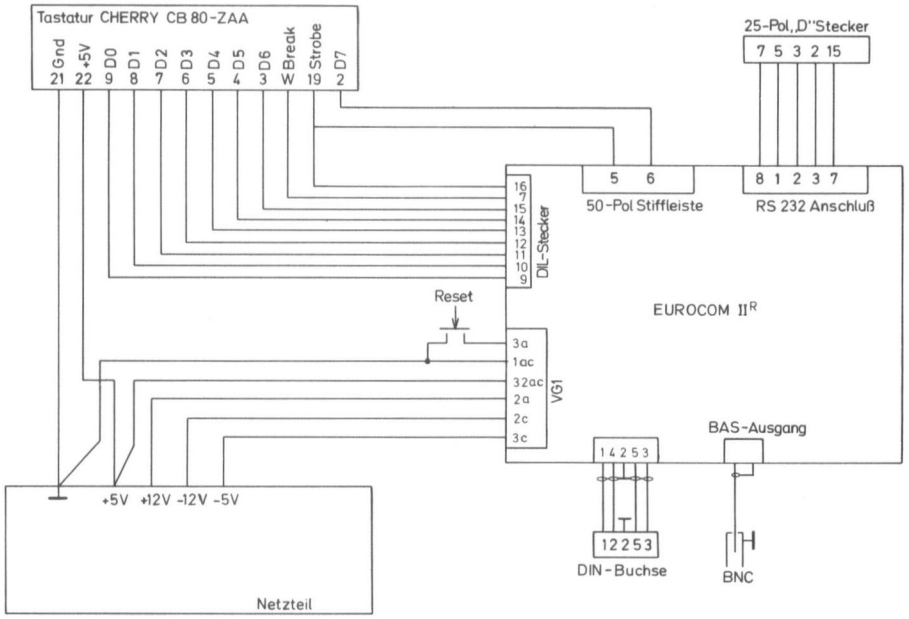
Der Überspannungsschutz besteht z.B. bei + 5 V aus D2, ZD1, R1 ··· R4,T1 und dem Transistor T2.

Übersteigt die Spannung am Ausgang des Reglers die Spannung D2 + ZD1 = 0,7V + 4 · 7 V = 5,4 V, schalten die Dioden




**Schaltung für ein µP-Netzteil, wobei hier die notwendigen Anforderungen für den EUROCOM II berücksichtigt worden sind**

**Verdrahtungsplan zwischen der Platine des EUROCOM II, dem Netzgerät und der Tastatur**




durch, die Darlington-Schaltung öffnet, und die Sicherung wird über R1 abgebrannt.
Die übrigen Spannungen sind entsprechend abgesichert, mit dem Unterschied, daß wegen der geringeren Ströme auf eine Darlington-Schaltung verzichtet werden konnte.
Kontrollschaltung (ZD2, R5, R15, C13, T6, D9, Rx, Relais): Wenn +5 V vorhanden sind, schaltet T6 mit einer durch R14, C13 bestimmten Verzögerung durch, und das Relais kann mit S2 eingeschaltet werden. Über das Relais werden die +5 V, +12 V und −12 V geführt, so daß die −5 V tatsächlich zuerst anliegen. Das Relais ist in Selbsthalteschaltung verdrahtet, dadurch wird erreicht, daß schon ein kurzer Ausfall der +5 V z.B. im Überlastfall zum Abschalten der drei Spannungen führt. Im Ausschaltmoment der Netzspannung unterschreitet die +5 V schnell die Abschaltschwelle von T6. Das Relais fällt ab, während die −5 V noch bis zur Entladung des Siebkondensators anliegen. Dadurch werden die −5 V zuletzt ausgeschaltet.
Die Verwendung dieses Netzgerätes beschränkt sich natürlich nicht auf den EUROCOM II, es kann vielmehr als universelles Mikrocomputer-Netzgerät angesehen werden.

## Der Aufbau

Alle Teile sind bis auf Netztrafo, Ladekondensatoren und zu kühlende Teile auf einer Platine untergebracht.
Als Netztrafo wird ein spezieller MP-Trafo verwendet, dessen Spannungen gut auf unseren Bedarf abgestimmt sind. Die Reserven dürften auch für größere Erweiterungen ausreichend sein; insbesondere wenn schließlich nur noch dynamische RAM mit **einer** Versorgungsspannung verwendet werden.
Als Kondensatoren werden Computer-Kondensatoren hoher Kapazität und Qualität verwendet.

Der Kühlkörper hat Mehrfachfunktion: Er dient als Rückwand und Befestigung für das Chassisblech, und er nimmt alle Regler sowie Gleichrichter und Netzbuchse mit Sicherung auf.

## Die Verdrahtung

Dieser Teil ist der aufwendigste und zugleich schwierigste des Netzteilaufbaus.

Zur Beachtung:
Die Farben der verwendeten Leitungen sollten sinnvoll gewählt werden, für jede Wechselspannung zwei gleiche Farben, Masseleitungen nur Schwarz usw. Alle Enden sind zu verzinnen.
Die gesamte Anlage besitzt nur einen Massepunkt, an den sämtliche Masseleitungen angeschlossen werden müssen. Nur so können Erdschleifen (Brummen und Schwingen) vermieden werden.
Es ist darauf zu achten, daß die Kabel möglichst kurz und niederohmig sind. Ich verwende für die +5 V eine 4-mm-Litze, für die übrigen Spannungen eine 1,5 mm flexible Meßleitung.

## Der Einbau der Tastatur

Am besten fertigt man sich gemäß Maßzeichnung eine Schablone aus Sperrholz oder Preßpappe. Man überzieht die Bedienschräge (nach Lösen von drei Schrauben nach vorn abnehmbar) mit selbstklebender Folie, um Kratzer zu vermeiden.

## Die Befestigung der Tastatur

Um Schrauben auf der Frontplatte zu vermeiden, werden zwei U-Profilleisten (Foto) mit eingeklebten Muttern M3 von hinten gegen die Bedienschräge geklebt. Die Tastatur wird daran verschraubt. Man hat etwas Spiel, so daß die optimale Position der Tastatur gefunden werden kann.

## Der Einbau einer Break-Taste

Eine Break-Taste (MC6809 NMI) in der Tastatur ist zwar bei CB 80-12AA vorhanden, aber diese ist nicht, wie es für den EUROCOM II wünschenswert wäre, getrennt als NMI-Signal herausgeführt. Ein Blick auf die Platine zeigt, daß dieses Signal an Pin W zur Verfügung stehen müßte, jedoch ein IC nicht bestückt ist. Wie sich feststellen läßt, ist ein SN74LS30 direkt einsetzbar. Man lege also mit Cu-Lack-Draht eine Verbindung Pin nach Pin, setze eine Fassung und ein LS30 ein, und schon hat man das gewünschte Signal an Pin W.

Jörg Lohse

# A/D- und D/A-Wandler für Sprachverarbeitung

Benutzer von Mikrocomputern mit RAM ab etwa 16 KByte haben im Prinzip die Möglichkeit, sich mit der Ein- und Ausgabe und Verarbeitung menschlicher Sprache zu beschäftigen. Meistens bleibt es nach Versuchen, bei denen das Kassettenrecorderinterface dazu zweckentfremdet wurde, wegen der daraus resultierten mangelhaften Sprachverständlichkeit bei einer Resignation, und es folgt die Aufgabe dieses interessanten und noch im Stadium der Forschung befindlichen Gebietes.

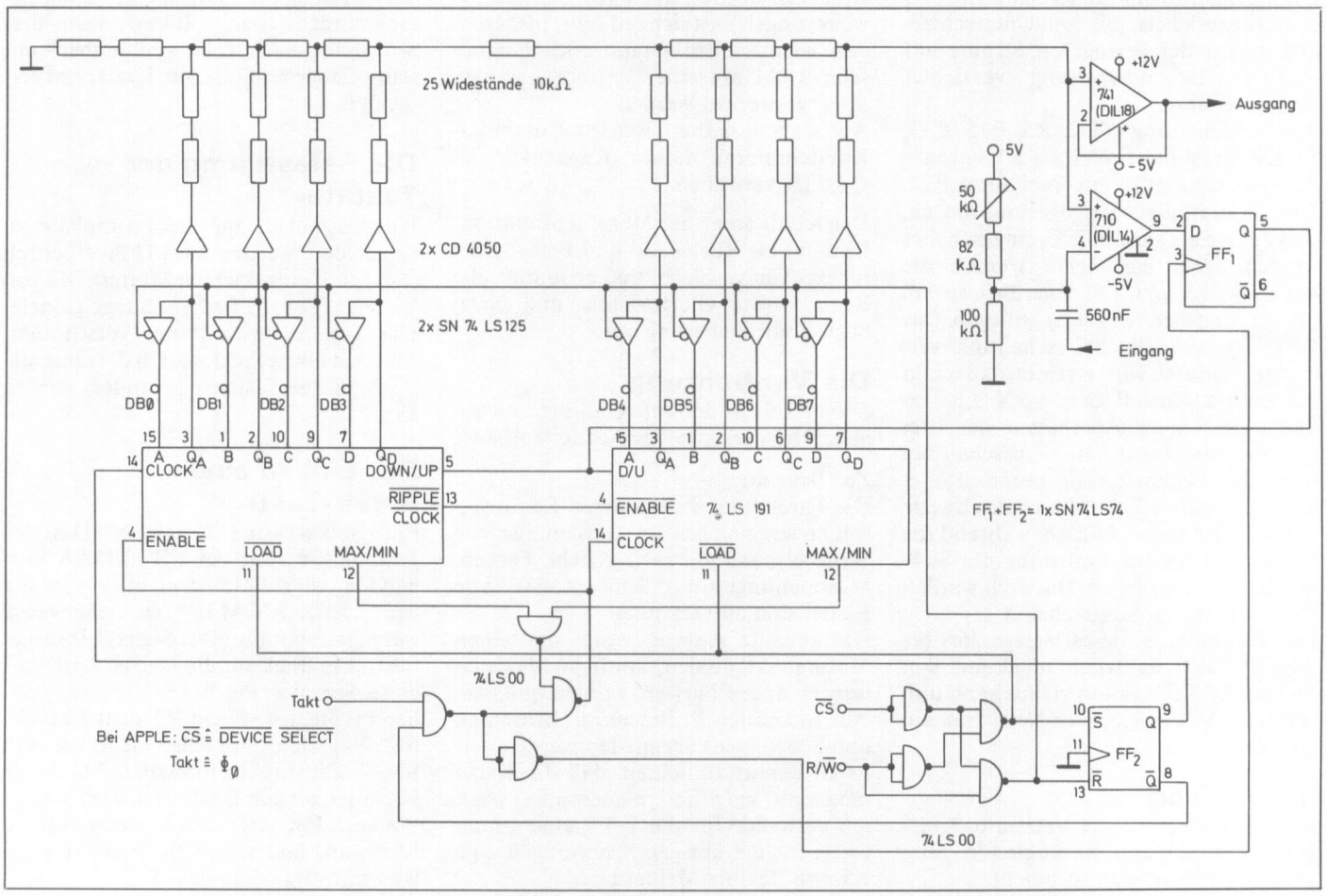
Denn zum ernsthaften Betreiben von Sprachverarbeitung werden schnelle 8-bit-D/A- und -A/D-Wandler benötigt. Der A/D-Wandler stellt die jeweilige anliegende Spannung des Sprachsignals als 8-bit-Wort zur Verfügung. Der D/A-Wandler wandelt ein 8-bit-Wort in eine entsprechende Spannung um.

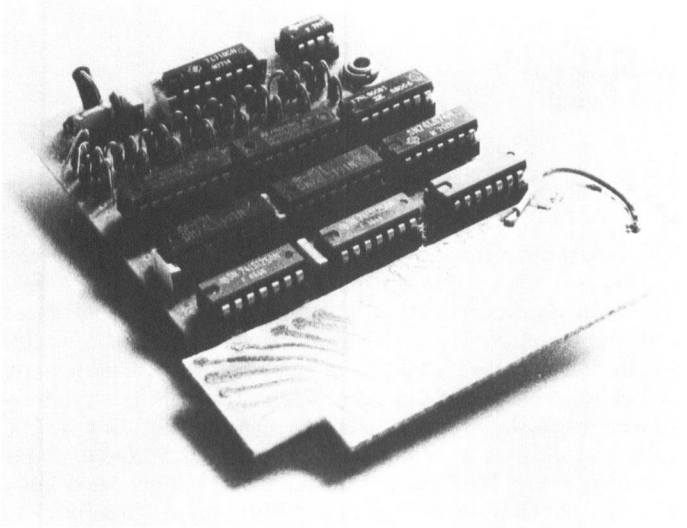
Hier wird eine zur Sprachverarbeitung geeignete Schaltung angegeben, die nur preiswerte und leicht beschaffbare Bausteine enthält.

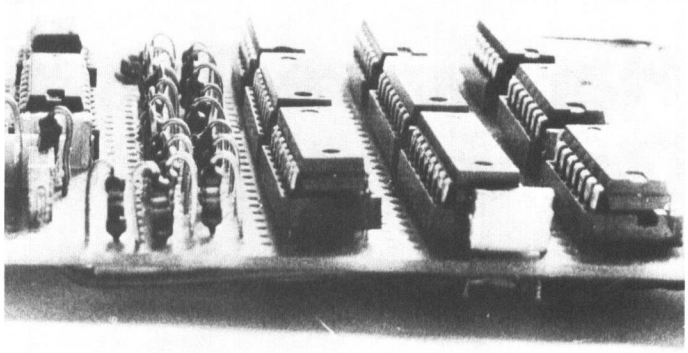
Die Schaltung wird vom Mikroprozessor wie ein Speicherplatz adressiert. Wird unter die Adresse der Schaltung geschrieben, so wird die Schaltung automatisch zum D/A-Wandler, und die dem geschriebenen Wert entsprechende Spannung ist am Analogausgang der Schaltung verfügbar. Liest der Mikroprozessor aus der Schaltung, so wird die Schaltung zum A/D-Wandler, und die gelesenen Byte entsprechen dem am Analogeingang der Schaltung anliegenden Momentanwert der Spannung. Wegen der automatischen Umschaltung der Betriebsart der Schaltung kann ohne zusätzlichen Softwareaufwand sehr komfortabel mit ihr gearbeitet werden.

Die Schaltung wird an den Datenbus des Mikroprozessors angeschlossen. Ferner benötigt sie ein Signal zur Bausteinauswahl ($\overline{CS}$) und das Read-Write-Signal (R/W) sowie ein Taktsignal von mindestens 1 MHz. Bei Systemen mit dem Mikroprozessor 6502 läßt sich das Taktsignal $\Phi_0$ (oft auch als $\Phi_2$ bezeichnet) benutzen. Die Anpassung an Systeme mit anderen Mikroprozessoren bereitet kaum Probleme. Es ist lediglich statt R/W ein gleichwertiges Signal zu benutzen, welches nur bei Schreibzyklen zu L-Pegel wird. Auch ein passendes Taktsignal wird sich immer finden lassen.

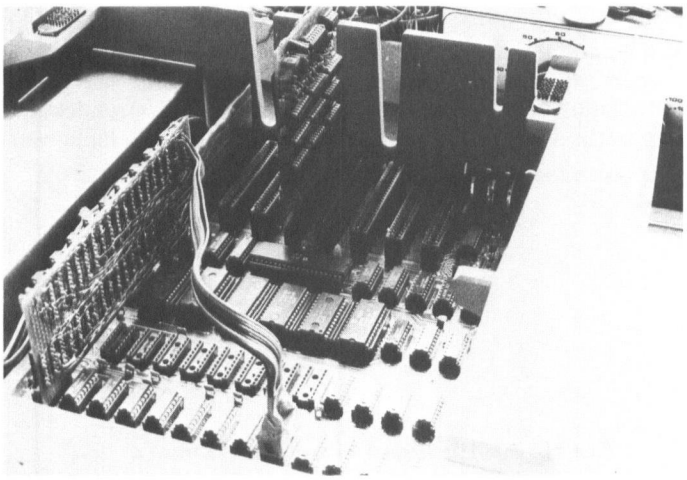
Die Musterschaltung ist für den Apple II (plus) konzipiert und wurde mit einem selbstgeätzten Stecker für die Steckplätze im Apple versehen. Dort wird als $\overline{CS}$-Signal das $\overline{\text{Device Select}}$-Signal verwendet. Die Schaltung kann mit der Hex-







**D/A-A/D-Karte**



**Hier ist der Apple mit dieser Schaltung in Steckplatz 3 zu erkennen. Im Steckplatz 0 ist eine selbstgebaute Language-Karte; damit stehen 64 KByte zur Sprachverarbeitung zur Verfügung**

Adresse C0n0 angesprochen werden, wobei n die Nummer des Steckplatzes plus 8 ist. Alle benötigten Signale und Spannungen sind an allen Steckplätzen vorhanden.

## Die Funktion der Schaltung

Es ist zum Schaltbild noch anzumerken, daß die Betriebsspannung von +5 V mit mindestens vier Kondensatoren von 100 nF nach Masse abzublocken ist. Die −5-V- und die +12-V-Spannungen sind mit je einem 100-nF-Kondensator nach Masse abzublocken.

Von dem Analogsignal gelangt der Wechselspannungsanteil auf einen Spannungsteiler und zum invertierenden Eingang des Komparators 710. Das Eingangssignal wird mit der vom Widerstandsnetzwerk erzeugten Spannung verglichen. Die Spannung des Widerstandsnetzwerks ist dem Zählerstand der beiden 5-bit-Zähler, die als Herz der Schaltung einen 8-bit-Vorwärts-Rückwärts-Zähler bilden, proportional. Wenn das Eingangssignal kleiner als das Referenzsignal vom Widerstandsnetzwerk ist, zählt der Zähler rückwärts. Ist das Eingangssignal größer, zählt er vorwärts. Die Referenzspannung des Netzwerks wird also immer der Eingangsspannung nachgeführt, und der Zählerstand ist damit ein Maß für die momentane Eingangsspannung. Liest der Mikroprozessor, so schalten die acht Tri-State-Puffer an den Ausgängen der Zähler durch, und der Zählerstand gelangt auf den Datenbus.

Das Signal des Komparators wird mit $FF_1$ zwischengespeichert, damit es sich während der aktiven Flanke des Taktes nicht ändert. Um ein weiteres Vorwärts- bzw. Rückwärtszählen bei maximalem bzw. minimalem Zählerstand zu unterbinden, wird das Taktsignal des Zählers in solchen Fällen gesperrt; anderenfalls würden schon bei leichten Übersteuerungen unerträgliche Verzerrungen entstehen.

$FF_2$ speichert die aktuelle Betriebsart. Wenn die Schaltung als D/A-Wandler arbeiten soll, bleibt das Taktsignal permanent gesperrt. Bei Schreibzyklen wird der Zähler mit dem geschriebenen Byte geladen, und der Zählerstand bleibt erhalten, bis er überschrieben wird. Der Zählerstand wird vom Netzwerk zur ihm proportionalen Spannung umgeformt, welche am Ausgang des als Impedanzwandler geschalteten Operationsverstärkers kurzschlußfest zur Verfügung steht.

Beim ersten Lesezyklus der als D/A-Wandler betriebenen Schaltung wird noch das zuletzt eingeschriebene Byte gelesen, die Schaltung aber gleichzeitig zur A/D-Wandlung umgeschaltet, so daß bei den nächsten Lesezyklen wieder die digitalisierte Eingangsspannung gelesen wird.

Die Ausgänge der Zähler sind über CMOS-Buffer an das Netzwerk geschaltet, da die CMOS-Logik geringere Abweichungen von idealen Schaltern zwischen Masse und +5V aufweist als TTL-Bausteine. Beim Nachbau ist zu beachten, daß nur acht von den zwölf CMOS-Buffern in den beiden 4050 verwendet werden. Die Eingänge der nichtverwendeten Buffer sind an Masse zu legen.

Die Eingangsspannung des Sprachsignals soll maximal 5 $V_{ss}$ betragen. Der Eingang ist also mit dem Lautsprecherausgang der Signalquelle zu verbinden.

Beim Nachbau der Schaltung sollte es keine Probleme geben. Beim Komparator 710 ist zu beachten, daß sich die im Schaltbild angegebenen Anschlußnummern auf das DIL-14-Gehäuse beziehen.

Der Einsteller ist so einzustellen, daß sich bei A/D-Wandlung und kurzgeschlossenem Eingang etwa 7FH oder 80H ergibt.

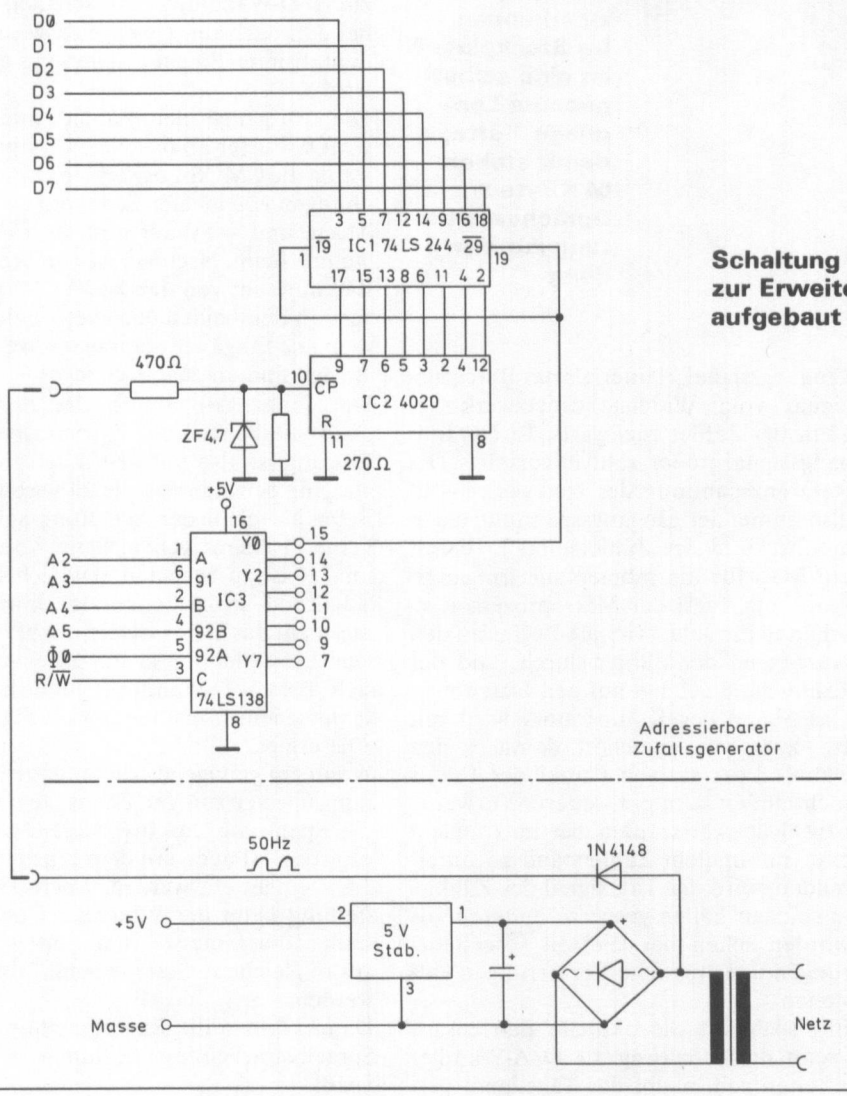
Wenn ein genügend hochohmiger Spannungsmesser zur Verfügung steht, kann die Spannung am invertierenden Eingang des 741 auch mit dem Einsteller auf 2,5 V eingestellt werden. Nach der Einstellung kann der Widerstand des Einstellers ausgemessen und durch einen etwa gleichen Festwiderstand ersetzt werden.

Dann können die Experimente mit der Sprachverarbeitung beginnen – viel Spaß!

Karlheinz Fiëtta

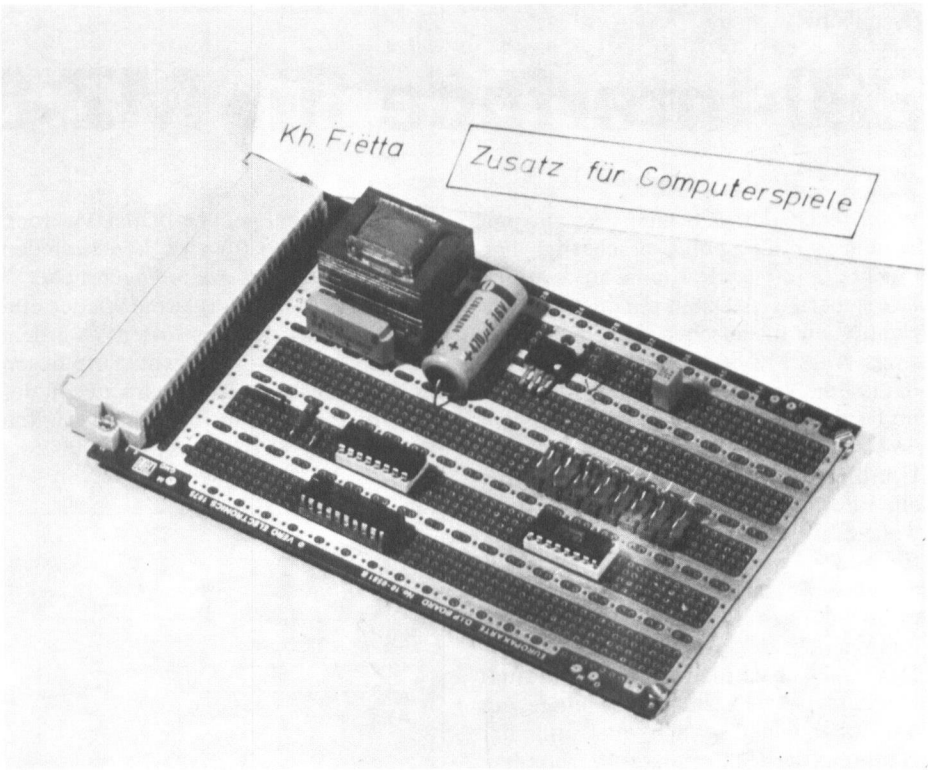
# Hardware-Zusatz für Computerspiele

Nicht alle Home- und Mikrocomputer bieten die Möglichkeit zur Erzeugung einer Zufallsgröße. Mit dem beschriebenen Zusatz wird es möglich, Glücksspiele zu programmieren, die vom Zufall beeinflußt werden können. Der dazu notwendige Zufallsgenerator liefert rein zufällige Werte an den Mikroprozessor.
Es ist zwar auch möglich, durch ein Unterprogramm zufällige Werte zu erzeugen, aber diese Werte weisen doch immer eine periodische Wiederholung auf, weil ihre Generierung durch den Systemtakt gesteuert wird. Erst ein durch ein unabhängiges Signal hervorgerufener Wert weist eine absolute Zufälligkeit auf. In der nachstehend gezeigten Schaltung läßt sich solch ein Zufallsgenerator mit nur drei Schaltkreisen aufbauen.
Für 8-bit-Prozessoren muß ein Byte erzeugt werden, dessen völlig zufälliger Wert über den Datenbus in den prozessorinternen Akkumulator geholt werden kann, um es von dort aus weiter verarbeiten zu können. Realisierbar ist dies mit Hilfe eines Zählers, der seinen Takt z.B. aus der Netzfrequenz bezieht. Im Schaltplan ist dies ein CMOS-Zähler aus 12 oder 14 hintereinandergeschalteten D-Flipflops. Die Flipflops teilen die Frequenz von Stufe zu Stufe herab und lassen so an den Q-Ausgängen eine sich ständig ändernde Bitkombination entstehen. Da von vorneherein der Zeitpunkt des Abfragens des 8-bit-Wortes nicht festliegt, ist der Wert des Byte nicht vorhersehbar, also rein zufällig. (Eine Voraussetzung muß allerdings noch erfüllt sein: Zwei aufeinander folgende Zufallswerte dürfen höchstens 0,04 s nacheinander erfragt werden. Der Grund: Das 50-Hz-Signal wird in der ersten Stufe auf 25 Hz geteilt, somit steht frühestens nach 0,04 s das nächste Wort zur Verfügung; ein Fall, der praktisch immer gegeben ist.)
Damit das Zufallswort auf den Systembus geschaltet werden kann, bedient man sich eines Bustreibers mit Tri-State-Ausgängen. Über die beiden Steueranschlüsse kann er vom Adressendekoder in den leitenden Zustand geschaltet wer-




**Schaltung für einen Hardware-Zusatz zur Erweiterung von Computerspielen, aufgebaut mit Standard-Schaltungen**

den. Der Dekoder ist mit den Adressenleitungen, dem R/W-Signal und dem $\Phi o$-Signal vom Systemtakt der CPU-Platine zu verbinden. (Bezeichnungen der Anschlüsse beziehen sich auf µP 6502, andere Prozessoren analog.)
Bei der Adressenwahl muß darauf geachtet werden, daß ein Adressenbereich gewählt wird, der noch nicht belegt ist. Damit die Auswahl leicht bleibt, lassen sich am Ausgang des 74LS138 acht verschiedene Adressen aufrufen. Eine Änderung der Zufallsgeneratoradresse ist mit dem Benutzen einer der Ausgänge Y1 bis Y7 möglich.
Das Abgreifen des 50-Hz-Signals am Netzteil bewerkstelligt man mit einer Diode D2 am Wechselspannungsanschluß des Gleichrichters. Die Bauteile R1 und D1 schneiden von den Sinushalbwellen die Werte über 4,7 V ab, damit sie von IC2 weiterverarbeitet werden können. Auf Flankensteilheit des Signals muß hier nicht geachtet werden, weil eine nicht exakte Signalform höchstens die Zufälligkeit des Signals an den Zählerausgängen erhöht. Bei der Wahl der Bauteile für IC1 und IC2 kann zwischen drei verschiedenen Typen gewählt werden.




**Platine mit der Schaltung für Computerspiele**

das aktuelle
ANGEBOT

**Sonderaktion**

**SNS - Nanocomputer-Sets**

**(Befristet bis 15. Dezember 1981)**

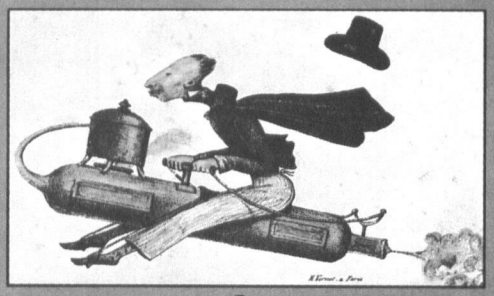
**Set 1: „Learning"**
1 Nanocomputer NBZ 80 S
1 Kassettenrekorder RCZ 80/P
1 Verbindungskabel W10Z80
1 Satz Bauteile K2Z80
1 Versuchssoftware NE-Z
1 Programmierblock
1 Zeichenschablone
Wert dieses Sets DM 2 452,– incl. Mwst
**Unser Aktionspreis** incl. MwSt **DM 1998,–**

**Set 2: „Hobby 1"**
1 Nanocomputer NBZ 80 B
1 Schachprogramm NCHESS
Wert dieses Sets DM 1 740,– incl. MwSt
**Unser Aktionspreis** incl. MwSt **DM 1194,–**

**Set 3: „Hobby 2 = Low Cost „Hobby 1"**
1 Nanocomputer NBZ 80
1 Schachprogramm NCHESS
Wert dieses Sets DM 1 338,– incl. MwSt
**Unser Aktionspreis** incl. MwSt **DM 698,–**

weisbauer elektronik
Heiliger Weg 1 · 4600 Dortmund 1 · Postfach 1645
Telefon (0231) 57 95 47 · Telex 822 538

**Technik**



**heiter**

Von der Tagnachtlampe zum fliegenden Dampfroß: Das große Schmunzelbuch vom Fortschritt mal ganz anders

Otmar Riedel (Hrsg.) **Technik Heiter**
17 x 24,5 cm. 336 Seiten. Gebunden. 28,– DM

**VOGEL-VERLAG, Fachbuchvertrieb**
**Postfach 67 40, 8700 Würzburg**

Uwe Köhl

# Z80-Interface für MZ-80 K

Wohl jeder, der sich eine Zeitlang mit seinem Minicomputer beschäftigt hat, möchte damit auch einmal andere Geräte steuern oder Daten mit anderen Einrichtungen austauschen. Dabei sind Besitzer eines MZ-80 nicht gut bedient, da in diesem Computer kein von außen verfügbarer Ein-/Ausgabeport eingebaut ist.

Um diesen Nachteil abzuhelfen, wurde ein Interface entwickelt und aufgebaut, welches für den MZ-80 (und damit auch für andere Z80-Systeme) eine parallele Schnittstelle mit $2 \times 8$ bit und eine serielle Schnittstelle zur freien Verwendung enthält.

Das Interface wurde in den freien Raum vor dem Netzteil des MZ-80 eingebaut und über ein Vielfachkabel mit der Hauptplatine des Computers verbunden. Dadurch ist die rückwärtige Steckleiste mit den Busanschlüssen weiterhin verfügbar. Alternativ kann das Interface natürlich auch über ein Vielfachkabel mit dieser Steckleiste verbunden werden, dann muß allerdings der Systemtakt und die Stromversorgung zusätzlich verkabelt werden.

## Die Vorarbeiten am Rechner

Nach dem Öffnen des Gehäuses (Netzstecker ziehen) werden die Steckverbindungen auf der Hauptplatine gelöst und seitlich oder nach hinten weggebogen. Nach dem Lösen der Verschraubungen kann nun die gesamte Platine herausgenommen und zur Seite gelegt werden. Dabei empfehlen sich die für das Arbeiten mit MOS-Bausteinen üblichen Vorsichtsmaßnahmen. Als nächstes wird nun ein Flachbandkabel (32adrig, Länge ca. 48 cm) an einem Ende auf etwa 10 cm in $2 \times 16$ Adern aufgeteilt, abgebogen und an diesem Ende mit zwei DIL-Stekkern versehen. Die andere Seite wird mit der Unterseite der Hauptplatine verlötet. In diesem Zusammenhang kann auch an der vorderen rechten Seite der Platine ein Resetschalter angebracht werden (ca. 20 cm Litze, Drucktaste in das vorhandene Loch an der Gehäuseunterseite).

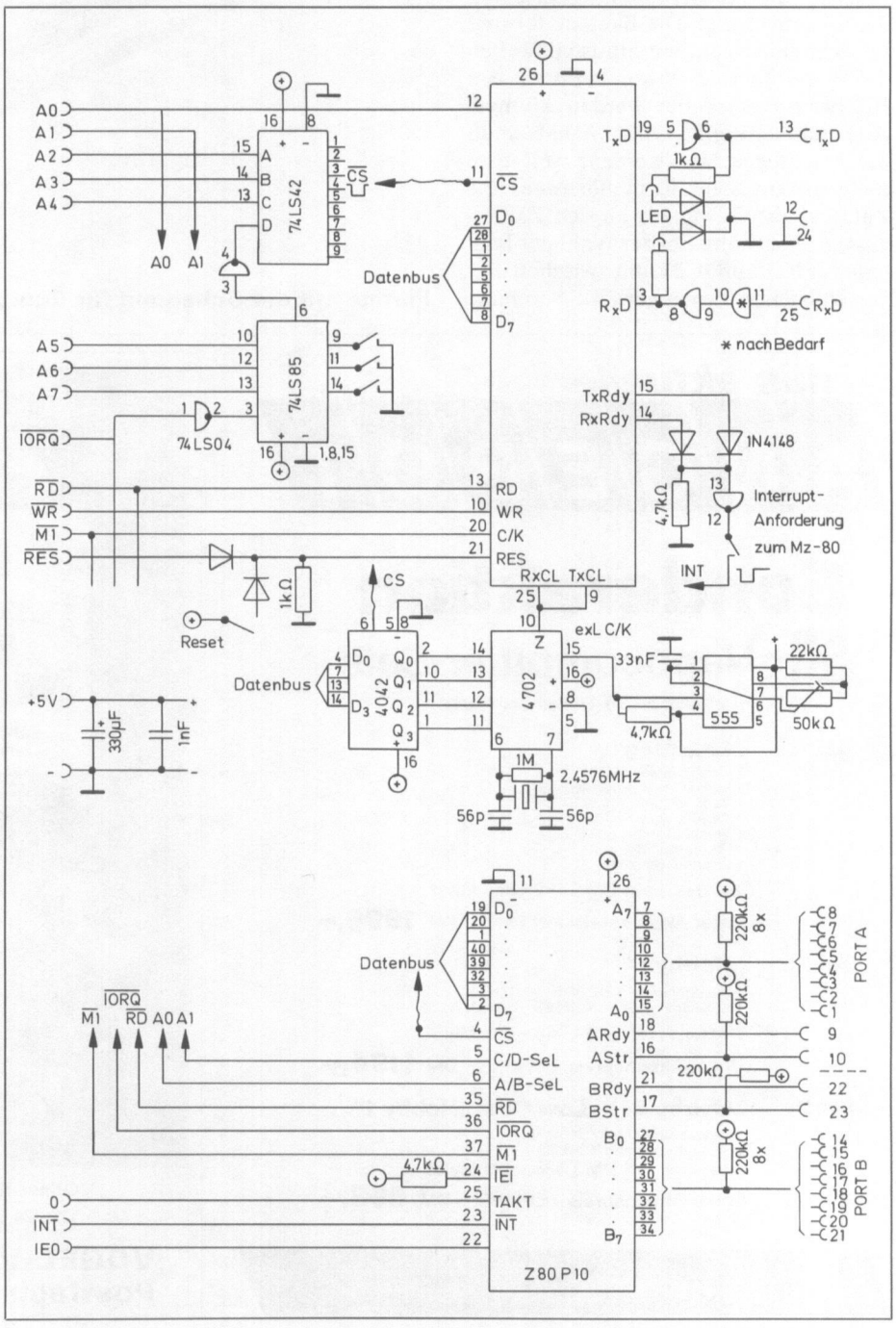
Nunmehr werden in eine Lochrasterplatte (ca. 100 mm × 120 mm) die Befestigungslöcher gebohrt und die Vielfachbuchse festgelötet. In der rechten Seitenwand der Bodenwanne werden entsprechend der Steckverbindung ein Ausschnitt angebracht und ebenfalls die Befestigungslöcher gebohrt. Zuletzt werden noch Bohrungen für zwei Leuchtdioden an der Gehäuseoberseite angebracht.

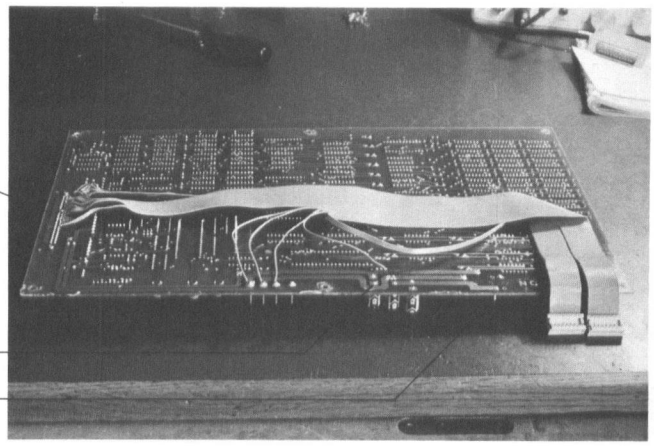
Jetzt kann die Hauptplatine wieder eingebaut und angeschlossen werden. Wurde alles richtig gemacht, dann liegen die zwei Stecker vorne rechts neben der Hauptplatine. Beim Wiedereinschalten des Rechners auf die Kurzschlußmöglichkeit durch die Stifte der DIL-Stecker achten.

Nun stehen der Steuer-, Daten- und Adressenbus sowie die Stromversorgung und der Takt zur Verfügung.

## Das Interface

Der Aufbau der Schaltung erfolgte in Fädeltechnik auf einer Lochrasterplatte.






**Unterseite der Hauptplatine mit den einzelnen Bezeichnungen**

Für alle Schaltkreise werden Fassungen verwendet, und alle Verbindungen sollten nach der Verdrahtung mittels Ohmmeter überprüft werden. Dabei werden, von den Fassungen von den DIL-Stekkern ausgehend, zuerst die Leitungen zu den IC-Fassungen durchgemessen und anschließend noch die interne Verbindung überprüft. Nun wird die Platte (ohne die IC) in das Gehäuse eingebaut und die Steckverbindungen zum Rechner hergestellt. Nach dem Einschalten des Rechners funktioniert dieser hoffentlich noch wie vorher. Nun wird die Stromversorgung der einzelnen IC nachgemessen (jeweils +5 V) und dann die Teile in der angegebenen Reihenfolge eingesetzt. Um Schäden an den Bauteilen zu vermeiden, ist dabei jedesmal der Computer auszuschalten.

## Der Timer 555

Der Timer erzeugt, zusammen mit dem 7402 und dem 4042, den Takt für den UART. Er ist zur Ergänzung des Baudratengenerators vorgesehen. Nach dem Einsetzen des IC steht an Pin 3 ein Rechtecksignal zur Verfügung. Die Frequenz richtet sich nach der gewünschten Baudzahl.

## Der Baudratengenerator 4702

Der Generator erzeugt die Frequenz für die gewünschte Baudzahl, ausgehend von einer Quarzfrequenz 2,4576 MHz. Je nach Programmierung liegt an Pin 10 die benötigte Frequenz. Für eine außerhalb der Reihe liegende Baudrate (z.B. 45,45 Baud) wird der Timer 555 eingeschaltet.

## 4 × D-Flipflop 4042

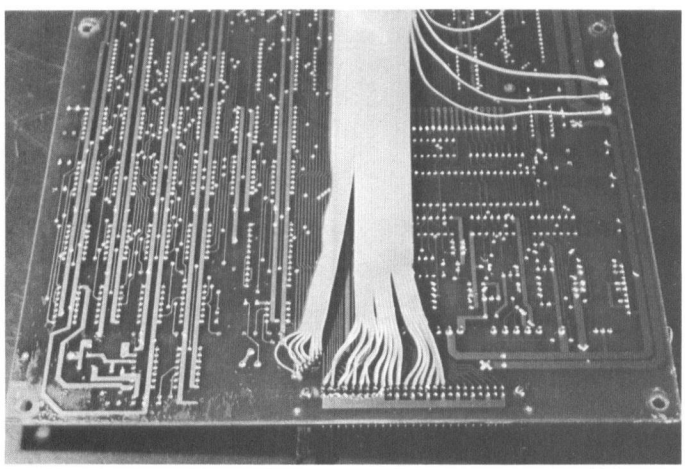
Über diese Flipflops wird der Baudratengenerator eingestellt. Die vier Flipflops speichern die gleichzeitig mit CS anliegenden Bit 1 bis 4 und stellen damit die Baudrate ein.

## Der Vergleicher 74 LS 85, Dekoder 74 LS 42, Inverter 74 LS 04

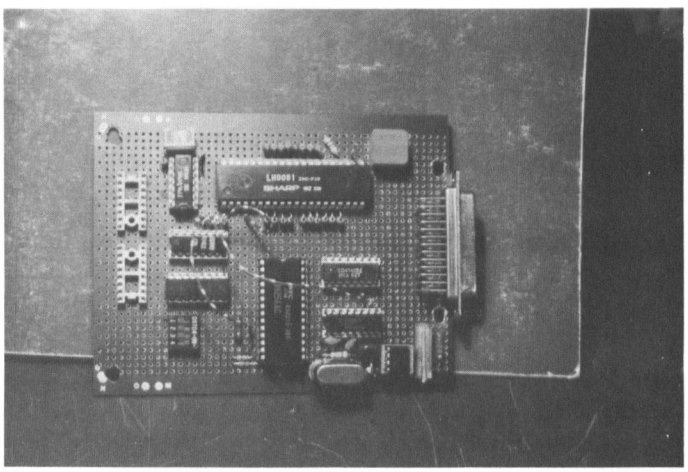
Diese Bauteile erzeugen die Chip-Selekt-(CS-)Signale für die zu programmierenden Bausteine über die Adressen A2 bis A7 und die Leitung IORQ (IN OR OUT REQUEST). Dabei werden über die DIL-Schalter 1 bis 3 und den Vergleicher die drei höherwertigen Bit sowie das Signal IORQ und über den Dekoder die drei nächsten Bit ausgewählt. Es können damit acht CS-Signale dekodiert werden. Über die Adressenleitungen A0 und A1 werden die Bausteine PIO und UART direkt angesprochen und Programmierung bzw. Datenein- oder -ausgabe festgelegt. Über zwei Inverter werden außerdem die Ein- und Ausgangssignale des UART gepuffert.

## Der Seriell-Parallel-Wandler (UART) 8251

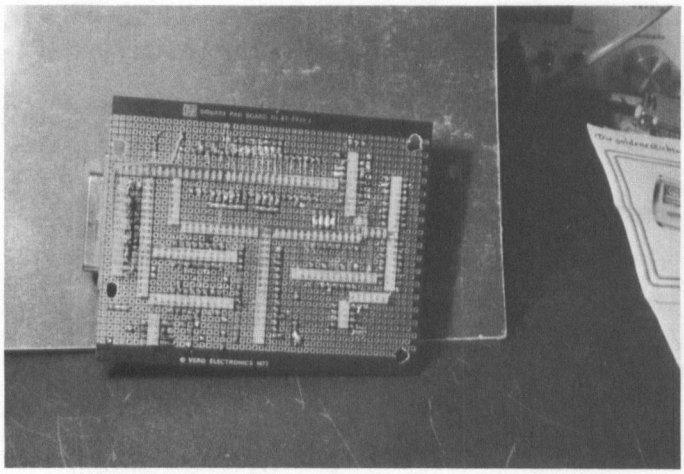
Dieser Baustein wandelt ein vom Computer angebotenes Signal (5 bis 8 bit) in ein serielles Signal um und umgekehrt. Die Daten werden über Pin 19 ausgegeben und über Pin 3 eingegeben und dabei gleichzeitig über zwei LED auch optisch angezeigt. Zur Pufferung ist jeweils ein Inverter (TTL) angeschlossen. Durch die Adresse A0 wird zwischen Steuerworten und Daten unterschieden, die Adresse A1 bleibt unberücksichtigt. Über die Ausgänge Pin 14 und 15 meldet der UART die Bereitschaft, ein neues Zeichen an den Computer abzugeben bzw. ein Signal vom Computer anzunehmen. Diese Ausgänge werden über ein Dioden-ODER-Gatter zusammengefaßt, invertiert und über einen Schalter an die Interruptleitung des Computers gelegt. Somit kann der UART nach Bedarf einen Interrupt anfordern.



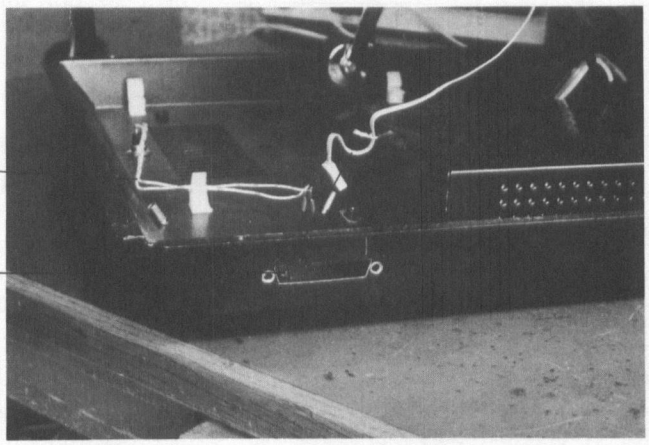
**Anschluß des Interface an der Computerplatine**



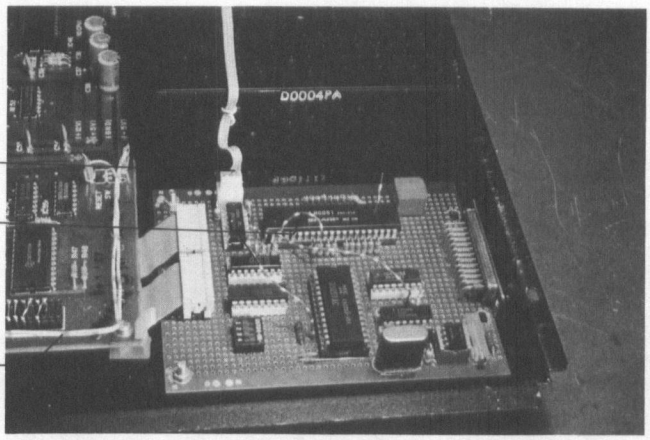
**Bestückungsseite der Interface-Platine**



**Fädelseite für die Interface-Platine**




**Geöffnetes Computergehäuse mit den Bezeichnungen**




**Eingebautes Interface mit den Anschlußbezeichnungen**

## Der Parallel-Input-Output (PIO) Z 80

Dieser Baustein kann 2 × 8 bit parallel ein- oder ausgeben, wobei jeder Port (A oder B) getrennt programmierbar ist. Außerdem können einzelne Bit ein- und ausgegeben werden und ein Interruptbetrieb durchgeführt werden. Durch die Adressen A0 und A1 wird zwischen Steuerung und Daten sowie Port A und B unterschieden.

### Baudrate/Programmierung

| Q0 | Q1 | Q2 | Q3 | Baud-rate | Steuer-wort |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | ext | 00 |
| 0 | 0 | 0 | 1 | ext | 01 |
| 0 | 0 | 1 | 0 | 50 | 02 |
| 0 | 0 | 1 | 1 | 75 | 03 |
| 0 | 1 | 0 | 0 | 134 | 04 |
| 0 | 1 | 0 | 1 | 200 | 05 |
| 0 | 1 | 1 | 0 | 600 | 06 |
| 0 | 1 | 1 | 1 | 2400 | 07 |
| 1 | 0 | 0 | 0 | 9600 | 08 |
| 1 | 0 | 0 | 1 | 4800 | 09 |
| 1 | 0 | 1 | 0 | 1800 | 0A |
| 1 | 0 | 1 | 1 | 1200 | 0B |
| 1 | 1 | 0 | 0 | 2400 | 0C |
| 1 | 1 | 0 | 1 | 300 | 0D |
| 1 | 1 | 1 | 0 | 150 | 0E |
| 1 | 1 | 1 | 1 | 110 | 0F |

z.B. für ext = 45,45 Baud
Clk. Freq. (Timer 555) = 718 Hz

| CS | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
|---|---|---|---|---|---|---|---|---|
| 0 | S3 | S2 | S1 | 0 | 0 | 0 | X | X |
| 1 | S3 | S2 | S1 | 0 | 0 | 1 | X | X |
| 2 | S3 | S2 | S1 | 0 | 1 | 0 | X | X |
| 3 | S3 | S2 | S1 | 0 | 1 | 1 | X | X |
| 4 | S3 | S2 | S1 | 1 | 0 | 0 | X | X |
| 5 | S3 | S2 | S1 | 1 | 0 | 1 | X | X |
| 6 | S3 | S2 | S1 | 1 | 1 | 0 | X | X |
| 7 | S3 | S2 | S1 | 1 | 1 | 1 | X | X |

A0, A1 sind für UART/PIO-Steuerung

**Beispiel**
S1, S2, S3 offen = log1
PIO = CS 1
Datenwort Port A
1 1 1 0 0 0 0 0 = E0
Datenwort Port B
1 1 1 0 0 0 0 1 = E1
Steuerwort Port A
1 1 1 0 0 0 1 0 = E2
Steuerwort Port B
1 1 1 0 0 0 1 1 = E3

Geschwindigkeitseinstellung

CS 3 Adresse
1 1 1 0 1 0 0 0 = E8 bis
1 1 1 0 1 0 0 1 = EB
UART = CS 4 Daten
1 1 1 0 1 1 0 0 = EC oder
1 1 1 0 1 1 1 0 = EE
Steuerung
1 1 1 0 1 1 0 1 = ED oder
1 1 1 0 1 1 1 1 = EF

Wolfgang Braun

# 220-V-Interface für Apple II

In der heutigen Zeit ist der Traum vom vollcomputerisierten Haus durch Einführung der Mikro- und Minicomputer in greifbare Nähe gerückt. Morgens wird man per Computer durch Musik geweckt, der Kaffee wird automatisch zubereitet, während man sich zur Morgentoilette im Bad aufhält, und wenn man das Eßzimmer betritt, geht automatisch das Licht an. Während der Urlaubsreise erscheint das Haus (die Wohnung) bewohnt, weil sich die Beleuchtung ein- und ausschaltet, und sich eventuell vorhandene Fensterläden automatisch öffnen und schließen.
All dies hat nur einen Haken, denn ein Gerät, das alle diese Funktionen per Computer steuert, ist bis jetzt noch nicht auf dem Markt. Um dem computergesteuerten Haus einen Schritt näherzugelangen, habe ich ein Interface entwickelt, das einen Apple II-Computer mit der Starkstromwelt verbindet.

```
]LIST
100  POKE 1009,76
110  POKE 1010,241
120  POKE 1011,165
130  POKE 1012,166
200  REM  ** DEMO PROGRAM FOR THE
     APPLE II INTERFACE BY
201  REM  ** WOLFGANG BRAUN
202  REM  ** HEINRICH DAUER STR.
     2
203  REM  ** 5160 DUEREN
204  REM  ** WEST GERMANY
205  REM  ** SEND QUESTIONS AND P
     ROGRAMMING TIPS TO ABOVE ADR
     ESS
210  REM  ** SET OUTPUTS HIGH
220  POKE  - 16295,0: POKE  - 162
     93,0: POKE  - 16291,0: POKE
      - 16289,0
300  REM  **SWITCHING ROUTINE
310  TEXT : HOME
320  HTAB 2: VTAB 1: PRINT "WHICH
     OUTPUT DO YOU WANT TO CHANG
     E ?"
325  HTAB 2: VTAB 3: PRINT "( TYP
     E 0 TO ESCAPE )"
330  HTAB 2: VTAB 6: INPUT "TYPE
     A NUMBER BETWEEN 1 AND 4 PLE
     ASE !";OP%
340  IF OP% < 0 OR OP% > 4 THEN 3
     30
345  IF OP% = 0 THEN 500
350  IF OP% = 1 THEN  POKE  - 162
     96,0: POKE  - 16295,0
360  IF OP% = 2 THEN  POKE  - 162
     94,0: POKE  - 16293,0
370  IF OP% = 3 THEN  POKE  - 162
     92,0: POKE  - 16291,0
380  IF OP% = 4 THEN  POKE  - 162
     90,0: POKE  - 16289,0
390  TEXT : HOME
400  HTAB 2: VTAB 12: PRINT "NOW
     OUTPUT ";OP%;" HAS CHANGED !
     "
410  FOR PAUSE = 1 TO 1000: NEXT
     PAUSE
420  GOTO 310
500  TEXT : HOME
510  VTAB 12: INVERSE : PRINT "EN
     D OF TEST !": NORMAL
520  END
]
```

**Software für das 220-V-Interface des Apple II**

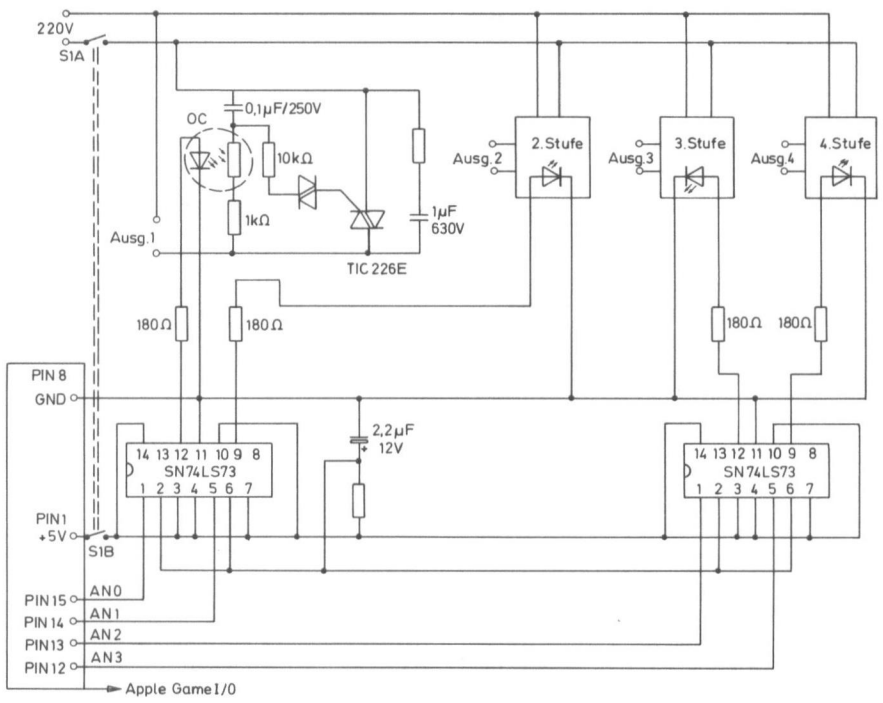
## Netzspannungsteil

Es lassen sich zur Zeit vier Verbraucher an mein Interface anschließen, von denen jeder maximal 1000 Watt Leistung verbrauchen darf. Das Ein- und Ausschalten der Verbraucher erfolgt mittels vier TRIAC, die per Optokoppler an den Apple II angeschlossen sind. Ich beschränke mich hier darauf, nur eine Stufe zu beschreiben, da die restlichen Stufen mit der ersten baugleich sind.

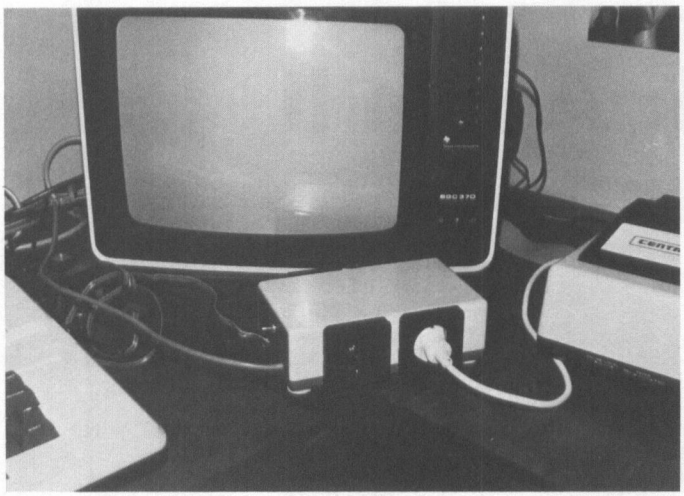
Wie aus dem Schaltbild zu ersehen ist, verwende ich einen TRIAC als Schalter, damit beide Halbwellen der Wechselspannung ausgenutzt werden. Dieser TRIAC wir über einen DIAC angesteuert. Als DIAC kann jeder herkömmliche Typ verwendet werden, solange er eine Schwellspannung von etwa 30 Volt besitzt. Diese Spannung reicht zum vollständigen Durchsteuern des TRIAC aus. Die Begrenzung des Steuerstroms erfolgt durch den 10-kΩ-Widerstand, der in Reihe zum DIAC geschaltet ist.

Der 4,7-Ω-Widerstand in Reihe mit dem 0,1-µF-Kondensator dient zur Glättung der Spannungsspitzen, die der TRIAC beim Schalten erzeugt. Eine weitere Entstörung ist nicht vorgesehen.

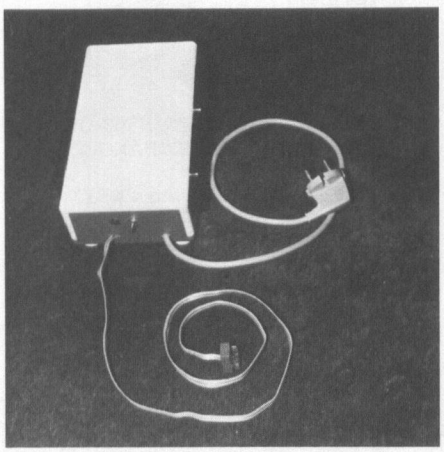
Der Rest der Schaltung ist, bis auf den Optokoppler, der im Schaltplan mit OC bezeichnet ist, eine Standard-Phasenan-



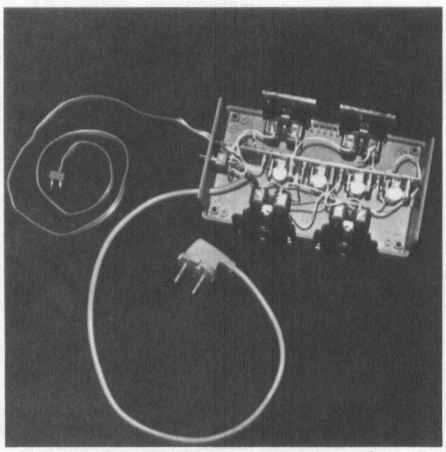

**Schaltung für das Interface mit zwei 74LS73 und Optokopplern**



**Aufgebaute Anlage mit Apple II, Monitor, Centronics-Drucker und selbstgebautem 220-V-Interface**



**Geschlossenes Interface-Gehäuse mit 220-V-Stecker**



**Geöffnetes Interface-Gehäuse und Anschlußkabeln**

schnittssteuerung und arbeitet nach dem Prinzip.

Der Optokoppler ist eine Eigenkonstruktion und besteht aus einer LED, die einen LDR (Fotowiderstand) belichtet. Beides ist in ein Plastikröhrchen eingebaut und lichtdicht abgeschlossen, damit sich die vier Stufen nicht gegenseitig beeinflussen.

Die LED ist eine rotleuchtende Diode, deren Anschlüsse mit Schrumpfschlauch isoliert sind, und die mit Zweikomponentenkleber an einem Ende des 10 mm (innen) durchmessenden Plastikröhrchens festgeklebt ist.

Am anderen Ende des Röhrchens wird ein LDR07 eingeklebt, und zwar mit der lichtempfindlichen Fläche in Richtung der LED. Diese Konstruktion verbindet den Apple II mit dem zu schaltenden Gerät und stellt gleichzeitig die Isolierung zwischen dem Apple und der 220-Volt-Netzspannung dar.

Der Rest des Netzspannungsteils wird von dem Schalter S1a gebildet, der die Netzspannung schaltet und dessen Teil S1b die 5-V-Spannung für die auf der Platine befindlichen Schaltkreise ein- oder ausschaltet.

## Niederspannungsteil

Das Niederspannungsteil der Schaltung besteht aus vier LED, vier Widerständen als Strombegrenzer bei den LED, einem RC-Glied und zwei Schaltkreisen.

Die vier 180-Ω-Widerstände dienen dazu, den durch die LED fließenden Strom mit Sicherheit unter 20 mA zu halten, damit die LED nicht beschädigt werden.

Das RC-Glied, bestehend aus einem 1-kΩ-Widerstand und einem 2,2-µF-Elektrolytkondensator, erzeugt beim Einschalten des Gerätes einen Clear-Impuls, damit man nach dem Einschalten immer von den gleichen Anfangsbedingungen ausgehen kann. Dies erleichtert die Programmierung ungemein, denn es besteht Apple-seitig keine Möglichkeit, den Zustand der Annunciator Outputs abzufragen.

Diese sog. Annunciator Outputs liegen am Game Paddle Anschluß des APPLE an den Pins 12 bis 15, und zwar entspricht Pin 15 dem AN0, Pin 14 ist AN1 usw. bis Pin 12. Näheres über diese Ausgänge findet sich im Apple II Reference Manual.

Diese Ausgänge steuern die beiden SN74LS73. Bei der Beschreibung dieses Schaltungsteils werde ich mich auch auf nur ein IC beschränken, da das andere die gleiche Funktion ausübt.

Das Innenleben des 7473 besteht aus zwei JK-Flipflops mit Clear, deren Ausgänge über die Optokoppler die Leistungsstufen steuern. Es wurde das SN74LS73 gewählt, weil es pulsgesteuert ist und Signaländerungen nur bei einer abfallenden TTL-Flanke erfolgen. Dies ergibt im Alltagsbetrieb eine hohe Störunanfälligkeit.

Der Nachteil ist, daß im Program zwei Befehle notwendig sind, um eine Änderung des Ausgangssignals zu erzielen.

## Softwarebeispiel

Zur Demonstration der Wirkungsweise des Interfaces habe ich ein kurzes Programm geschrieben, das im Anhang gelistet wird.

Die Zeilen 100 bis 130 setzen vier Flags im Apple, die beim versehentlichen Drücken der Reset-Taste das System neu booten.

Zeile 220 setzt alle vier Ausgänge am Game-Connector auf H-Pegel, damit definierte Anfangsbedingungen beim Start des Programms herrschen.

Danach wird über den Bildschirm abgefragt, welcher Ausgang geschaltet werden soll. Die Pulserzeugung für die 7473 erfolgt in den Zeilen 350 bis 380, wobei die Zahlen 1 bis 4 den jeweiligen Ausgängen entsprechen.

Zeile 400 zeigt an, welcher Ausgang verändert und welches Gerät geschaltet wurde. Die Software für diese Interface läßt sich sehr einfach erstellen, wie man aus meinem Beispiel ersehen kann. Für Funktionen, die zeitrelevant sind, wie z.B. das morgendliche Wecken per Computer, bietet sich der Einbau einer Uhr, die als Zusatz in einen der Peripheral-Slots gesteckt werden kann, an, oder man schreibt ein Uhrenprogramm.

Beat Stocker

# Tastenentprellung und Autorepeat für das Video-Genie-System

## Allgemeines zur Tastatur

Wer schon intensiver mit dem Video-Genie gearbeitet hat, der kann ein Lied davon singen. Ein großes Programm etwa in den Computer zu bringen ist eine echte Geduldsprobe.

Zwar macht sich das Problem der Preller vorerst beim Eintippen nicht so stark bemerkbar, weil man sie gerne übersieht. Spätestens bei der Programmausführung gehen dem fleißigen Tipper dann aber die Augen auf. Eine wahre Lawine von „Syntax Error-Meldungen" purzelt auf ihn zu. Häufigster Grund: Doppelte Buchstaben, z.B. FOOR, 8//2, POKE (16000, ⊥ 180). Dabei sind alle Tasten etwa gleich beteiligt, mit einigen Ausnahmen. Bei mir waren das z.B. die Tasten „Space", „L", „/".

Zwar sind die oben erwähnten Fehler vermeidbar, indem man sie sofort korrigiert, wenn man sie entdeckt. Eine mühsame Arbeit! Aber eben – wer kann schon blind schreiben und alles auf dem Bildschirm überwachen? Vielmehr verläßt man sich auf die Tastatur (vor allem bei längeren Satzgebilden) und wirft ab und zu einen Kontrollblick auf den Monitor. Dabei entdeckt man natürlich nicht alles, da es eben **nur** ein **Kontrollblick** ist.

Eine Computertastatur darf keine Prellerscheinungen haben. Alles andere ist Spielzeug.

Wer dem entgegenhält, daß der Video-Genie dafür preiswert ist, hat zwar recht, aber die Entprellung bei diesem System ist gar kein Preisproblem. Um dies zu verstehen, muß man wissen, wie die Tastatur beim Video-Genie arbeitet. (Es soll hier ausdrücklich erwähnt werden, daß die Arbeitsweise nur bei diesem System und dem TRS80 so ist. Der Commodore PET oder CBM funktionieren anders.)

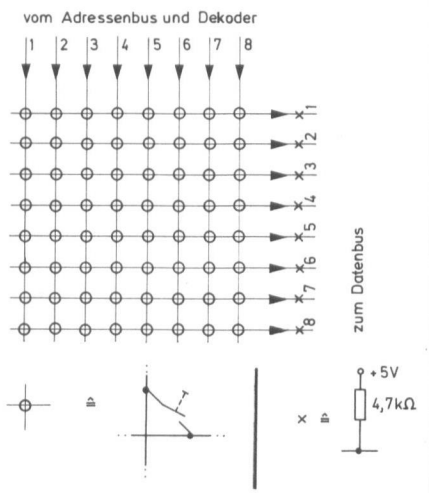
## Was sind Preller?

Preller entstehen hauptsächlich an mechanischen Kontakten. Da diese z.B. beim Einschalten nicht einfach „Null-Komma-plötzlich" schließen, entstehen unerwünschte Impulse für die Digitalschaltungen. Diese Tatsache ist mechanisch bedingt. So kann es vorkommen, daß man z.B. einer Zählschaltung einen Impuls mit Hilfe eines Drucktasters geben will, diese aber zehn registriert. Der Grund liegt in der Schnelligkeit, mit der Digitalschaltungen arbeiten.

Diese Erscheinung tritt nicht jedesmal auf, denn sie hängt vor allem von zwei Kriterien ab:

- Prellzeit einer Taste?
- Wo befindet sich der Computer mit der Tastaturabfrage?

Damit kommen wir nun zur Funktionsweise der Tastatur beim Video-Genie.




**Ansteuerung der Tastatur**

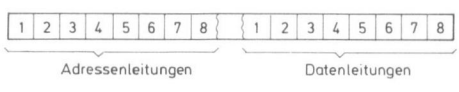
## Wie funktioniert die Tastatur?

Die Struktur der Tastatur ist ähnlich wie beim Taschenrechner. Wie aus dem Memory-Map von CHIP, Juli 1979 ersichtlich ist, belegt die Tastatur die Speicherplätze 14 336 – 15 359. Der Prozessor fragt nun diese Adressen über den Adressenbus ab, und je nachdem, ob eine Taste gedrückt ist, erscheint eine entsprechende Information auf dem Datenbus. Diese Information wird dann per Software in den ASCII-Zeichenkode umgewandelt. Die Tastatur stellt also eine direkte Verbindung zwischen Adressenbus und Datenbus dar. Auf die Tastatur bezogen heißt das, daß zuerst die Reihe 1 aktiviert wird (oben). Falls nichts gedrückt wurde, ist an den Spalten (rechts) überall eine „1". Diesen Pegel erzeugt der 4,7-kΩ-Widerstand. Ist aber eine Taste betätigt, so wird die „1" vom Abfrageimpuls auf „0" heruntergezogen. Dieses Bitmuster, das dabei auf dem Datenbus entsteht, wird nun vom Computer ausgewertet.

Adressenleitungen

| Datenleitungen | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | | Shift | | | | | | |
| 2 (Normal →) | Clear | New L. | Break | Back S. | Space | ESC | Tabul. | CTRL |
| 2 (Shift →) | Clear | New L. | Break | Shift BS. | Space | | CHR $ (23) | |
| 3 | 9 | 8 | : | - | / | , | · | , |
| | ) | ( | * | = | ? | + | > | < |
| 4 | 1 | 0 | 2 | 5 | 7 | 3 | 6 | 4 |
| | ! | Space | „ | % | ' | # | & | $ |
| 5 | y | x | z | ] | — | [ | ^ | \ |
| | y | x | z | ] | — | [ | ^ | \ |
| 6 | Q | P | R | U | W | S | V | T |
| | Q | P | R | U | W | S | V | T |
| 7 | I | H | J | M | O | K | N | L |
| | I | H | J | M | O | K | N | L |
| 8 | A | @ | B | E | G | C | F | D |
| | A | @ | B | E | G | C | F | D |

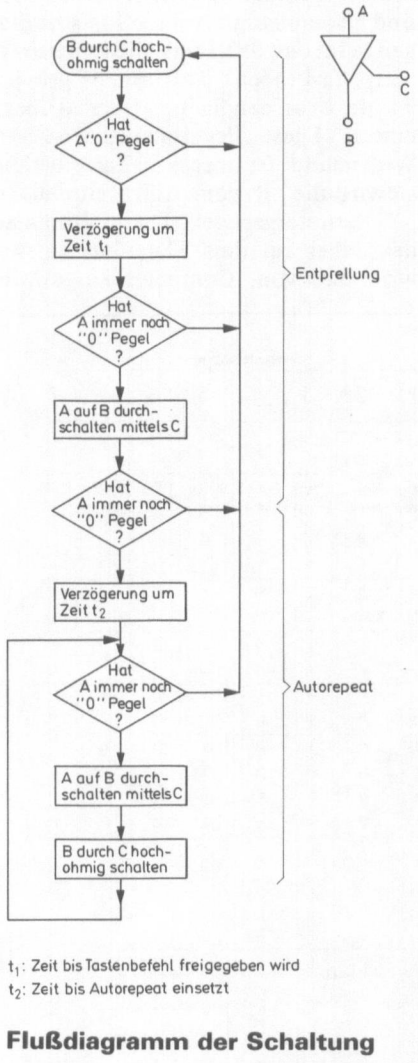
**Tastaturtabelle für die Anschlußklemme des Tastaturprints**



Zum Schluß erscheint dann das gedrückte Zeichen auf dem Monitor. Danach wird die Abfrage bei der nächsten Reihe fortgesetzt usw.
Die Verknüpfung der verschiedenen Tasten wurde durch die Tastaturplatine realisiert. Will man nun wissen, welcher Kreis welche Taste ist, so stellt man die Verbindung von dem Adressenbus zum Datenbus künstlich her. Dies geschieht im einfachsten Falle mit Hilfe eines Drahtes. Es werden die linken acht Eingänge mit den rechten acht Ausgängen auf dem Tastatursockel verknüpft. Das Ergebnis dieser Arbeit ist in der Tastaturtabelle festgehalten. Zwei Dinge fallen dabei auf: Zum einen könnten alle Zeichen erzeugt werden, wenn man dafür mehr Tasten auf das Keyboard montiert hätte. Diese müssen jetzt eben mit CHRS(xxx) erzeugt werden. Zum anderen ist die Funktion „CLEAR" des TRS 80 ebenfalls vorhanden, schlummert jedoch vor sich hin, weil ebenfalls keine Taste dafür vorgesehen worden ist. Eigentlich schade!

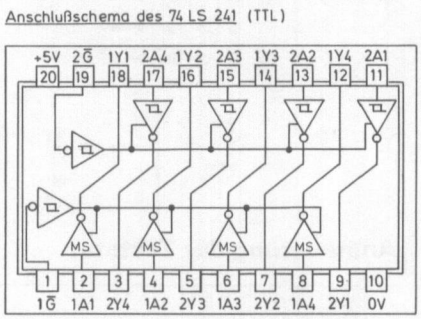
## Schaltungsbeschreibung

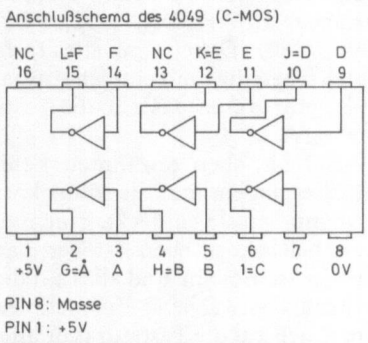
Das Flußdiagramm der Entprellschaltung soll parallel zum Schema erklärt werden. Die Schaltung wird zwischen den Ausgang des Tastaturprintsockels und der Rückleitung zur CPU-Platine eingeschoben. Zunächst wird keine Taste gedrückt. Dadurch kommt eine „1" über R1 an den Eingang von N1. Die Dioden sperren je in diesem Falle. Am Ausgang von N1 erscheint eine „0", die bewirkt, daß IC1 alle Ausgänge hochohmig hat. (Eingang 1 „1", Eingang 19 „0".) Mit den als Kreuz gezeichneten Widerständen werden die Ausgänge auf „1" angehoben. Dieser Zustand auf den Datenleitungen signalisiert dem Prozessor, daß nichts gedrückt ist. Der rechte Teil der Schaltung ist vorerst unwichtig, außer, daß C2 am rechten Anschluß auf „0" liegt.
Wird nun eine Taste gedrückt, so passiert folgendes: Über eine der Dioden D1 – D8 (je nachdem, welche Taste gedrückt wird) gelangt der Abfrageimpuls mit 0-Signal an den Eingang von N1. Da R1 sehr hochohmig ist, wird er jetzt nicht berücksichtigt. Zugleich liegt der Abfrageimpuls auch an dem IC1 (an einem der Eingänge der Puffer).
(Was nun nachfolgend beschrieben wird, wäre Aufgabe der Software gewesen. Offenbar sind die alten ROM des TRS 80 verwendet worden, denn bei den neuen wurde dieses Problem implementiert.)
Er wird aber vorerst noch nicht durchgeschaltet, da die Eingänge 1,19 die Ausgänge der Puffer noch hochohmig halten. Gleichzeitig lädt sich nun C2 über R2 auf. Wird die Spannung über C2 $\approx U_B/2 = 2{,}5$ V, so wechseln Gatter N5 und N6 ihre Zustände. Jetzt kann IC1 die Zustände der Eingänge auf die Ausgänge weitergeben. Die mit einem Kreuz gekennzeichneten Widerstände können auch hier vernachlässigt werden. Ein 0-Signal am Ausgang zieht die +5 V des Widerstandes ebenfalls auf „0" herunter. Mit der Verzögerung des Signals erreicht man, daß die Prellzeit der Tasten ignoriert wird. Die Signale gehen erst an die CPU, wenn die Taste endgültig geschlossen hat. Diese Zeit $t_1$ kann mit R2 einge-



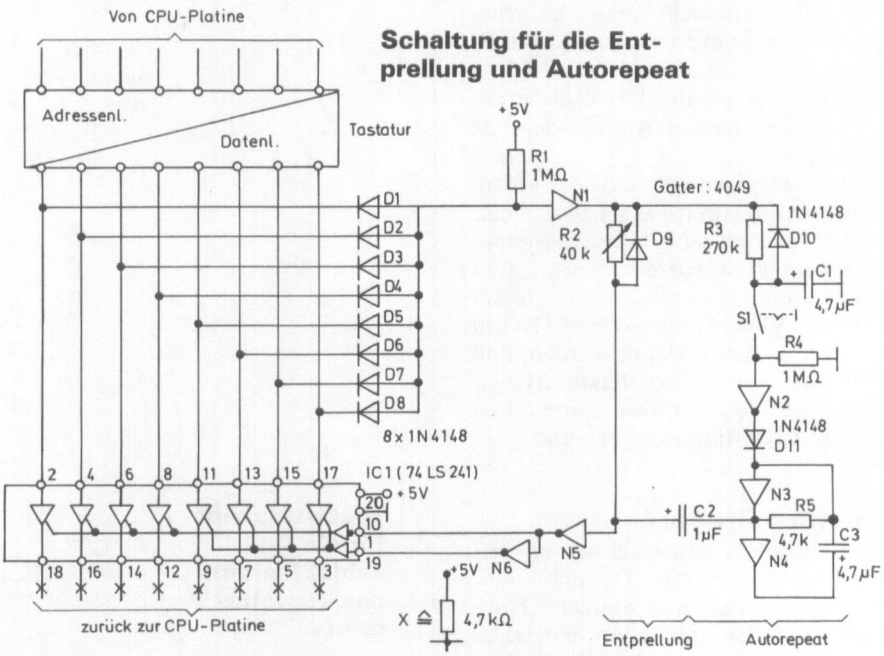

$t_1$: Zeit bis Tastenbefehl freigegeben wird
$t_2$: Zeit bis Autorepeat einsetzt

**Flußdiagramm der Schaltung**

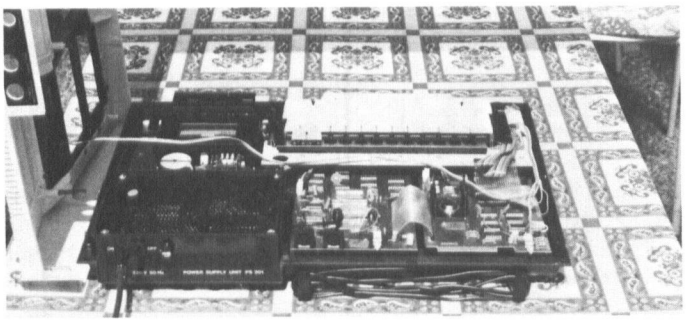



**PIN-Belegung der verwendeten Schaltkreise**

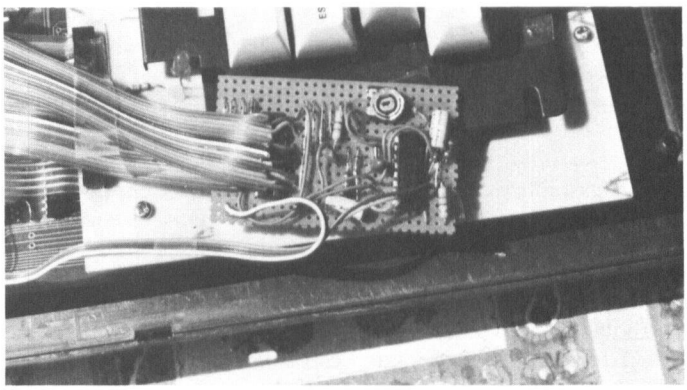


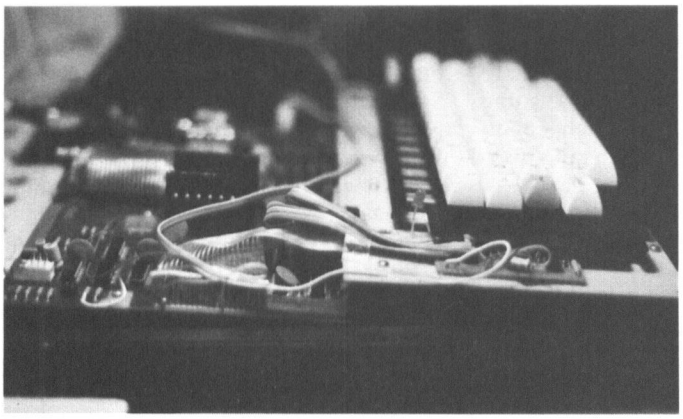

**Schaltung für die Entprellung und Autorepeat**



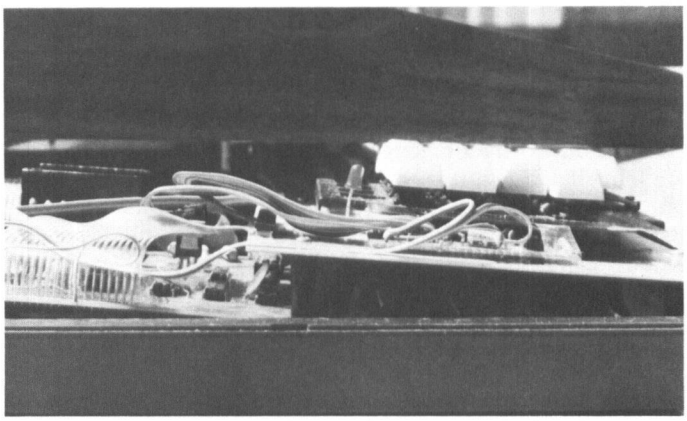
**Aufgeklapptes Video-Genie**

**Platine für die Tastenentprellung und Autorepeat ▸**





**Eingebaute Platine und Tastatur des Video-Genie**



**Nach dem Einbau wird das Gehäuse wieder fest verschraubt**

stellt werden. Sie darf nicht zu klein sein, weil sonst die Taste immer noch prellt, aber auch nicht zu groß, da dann die Ansprechzeit der Taste zu lang wird, und ein vernünftiges Schreiben unmöglich wird. Am besten ermittelt man sie durch ein Experiment [$t_1$ = f (R2, C2)].
Wird die gedrückte Taste wieder losgelassen, so entlädt sich C2 sofort über D9, da diese nun leitend ist. Dadurch werden die Ausgänge von IC1 hochohmig.
Bleibt sie aber weiterhin noch gedrückt, so lädt sich auch C1 soweit auf, daß der Ausgang von N2 „0" wird. Diese Zeit $t_2$ wird durch R3 und C1 bestimmt [$t_2$ = f (C1, R3)].
Auch hier wird C1 sofort entladen, wenn die Taste losgelassen wird, über D10. Nachdem N2, also „0" am Ausgang hat, sperrt die Diode D11. Dadurch kann der Taktgenerator, bestehend aus N3, N4, zu oszillieren beginnen. Durch den Ausgang von N3 wird nun C2 ständig entladen und wieder geladen. Damit setzt der Autorepeat ein, bis die Taste wieder losgelassen wird, da N5 bzw. N6 abwechselnd immer eine „0" und ein „1" am Eingang haben. Dies ist gleichbedeutend, als ob die Taste immer gedrückt und wieder losgelassen würde. IC1 schaltet abwechslungsweise durch oder sperrt. Mit S1 kann der Autorepeat abgeschaltet werden. R4 sorgt dann dafür, daß N2 nicht in der Luft hängt, was ja bei der CMOS-Technik verboten ist, und immer eine „0" erhält. Gleichzeitig wird dann der Ausgang „1" und der Oszillator wird über D11 blockiert. Damit hat C2 immer eine „0" am rechten Anschluß. Die Oszillatorfrequenz hängt von C3 und R5 ab [f = f (C3, R5)]. Die Werte dieser und aller anderen zeitbestimmenden Bauteile (R2, C2, R3, C1, R5, C3) sollen als Richtwerte gelten und können in bestimmten Größen variiert werden. R5 sollte aber nicht zu klein gewählt werden, da sonst ein unnötig großer Kurzschlußstrom durch D11 fließt. N3 hat ja immer das inverse Signal von N2, falls D11 nicht sperrt (immer wenn N2 eine „1" am Ausgang hat).

## Der Aufbau

Die Schaltung wurde auf einer Veroboard-Platte aufgebaut. Dank der geringen Anzahl von Bauelementen findet sie unter dem Tastaturblech gut Platz. Die Verbindungen von und zum Tastaturprint wurden mit Flachbandkabel ausgeführt. Am Ende der beiden Kabel wurde in Längsrichtung ein aufgeschnittener 16poliger IC-Sockel angebracht. Jenes Kabel, das von der Tastatur kommt, kann gesteckt werden auf beiden Seiten, das andere wird verlötet und mit dem Flachbandkabel des Computers zusammengesteckt.

Schalter S1 wurde mittels einer Bohrung durch das Gehäuseoberteil neben dem Tonbandzählwerk befestigt. Da S1 ein Schalter mit drei Stellungen ist, wurden zwei für Autorepeat EIN/AUS benötigt. Die dritte wurde zur Erzeugung der CLEAR-Funktion verwendet. Dazu muß einfach Reihe 1 und Spalte 2 auf den Schalter geführt werden.

Die Betriebsspannung kann an jedem TTL-IC abgegriffen werden (PIN 8 oder 7 Masse, PIN 14 oder 16 +5 V: stimmt in den meisten Fällen).

Auf den Fotos sind der Aufbau des Prints und die Kabelführung ersichtlich. Es empfiehlt sich, die Flachbandkabel möglichst kurzzuhalten, während die übrigen Verbindungen genügend Reserve haben sollten, damit das Oberteil des Computergehäuses noch immer demontiert werden kann. Das Computergehäuse selber löst man mit den Schrauben an der Unterseite des Computers. Es müssen dabei auch die Tastenkappen der Hinterseite (Reset, Video Cut) entfernt werden durch einfaches Abziehen. Danach wird das Oberteil des Gehäuses vorsichtig nach oben gezogen und nach vorne weggefahren. Aufpassen auf die Knöpfe der Hinterseite!

So, und nun viel Spaß beim Tippen, prellfrei, versteht sich!

Dieter Stöhlein

# EPROM-Programmiergerät für Z80

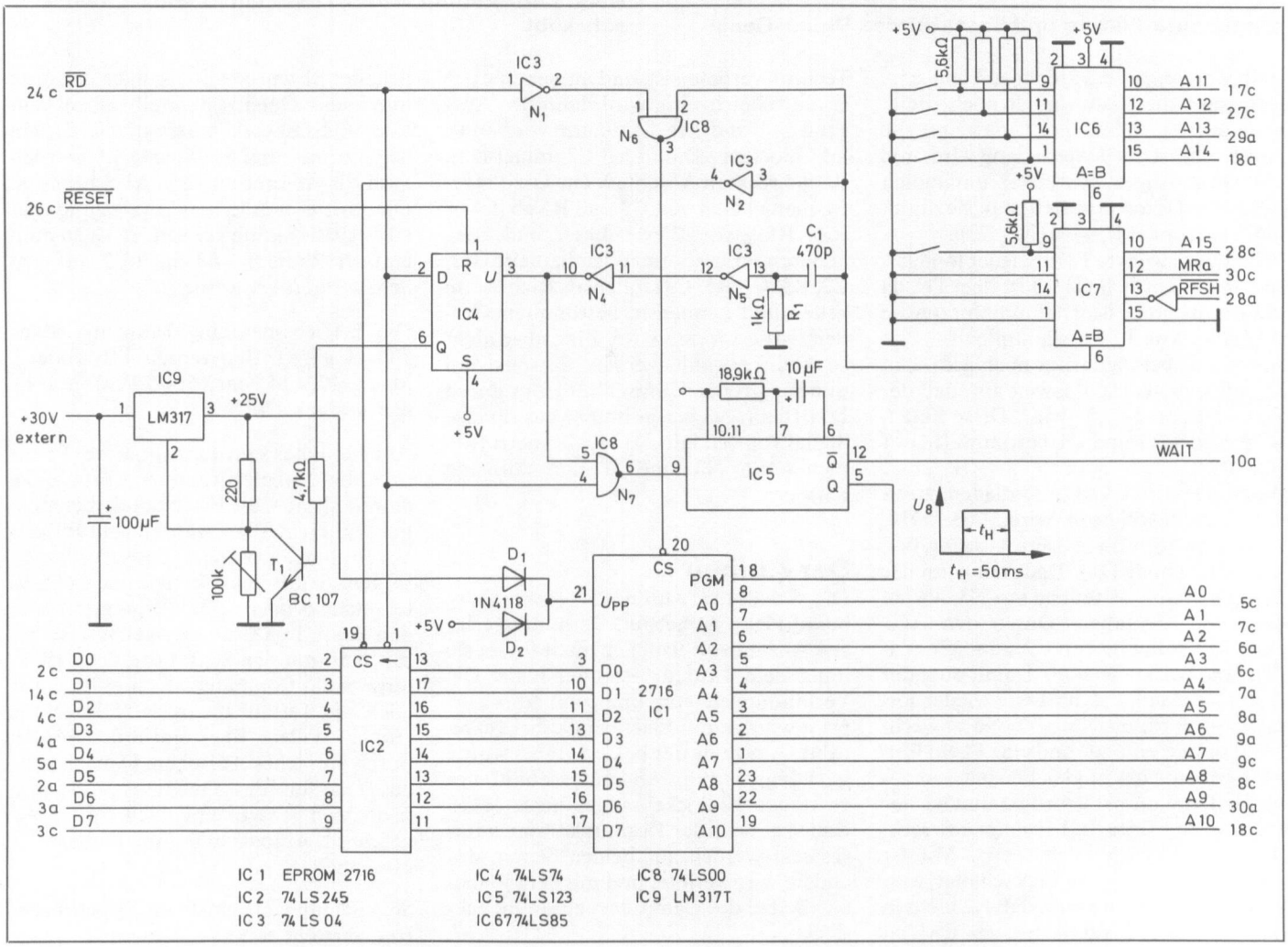
Die vorgestellte Schaltung stellt ein Programmiergerät für EPROM-Bausteine vom Typ 2716 dar. Die Schaltung besteht aus nur wenigen TTL-Bausteinen und arbeitet in Verbindung mit einer Z80.
Grundlage dieser Schaltung ist die Tatsache, daß bei einem Zugriff der Z80-CPU auf externe Speicher oder I/0-Einheiten und aktiviertem WAIT-Eingang, die CPU solange mit stabilen Daten- und Adressenbus verharrt, bis der WAIT-Eingang inaktiv wird, um dann weiter zu arbeiten.
Aufgabe des WAIT-Einganges ist es also, die schnelle CPU mit langsameren Peripheriebausteinen zu synchronisieren. Beim Programmieren stellt das EPROM einen solchen langsamen Baustein dar, da das Programmieren eines Byte etwa 50 ms erfordert.
Da ein EPROM keine Schreibleitung wie ein RAM-Speicher besitzt, muß eine Umschaltung der Datenleitungen von Lesen auf Schreiben anders realisiert werden. Durch Anlegen einer Gleichspannung von +25 V an den $U_{pp}$-Eingang des EPROM werden dessen Datenleitungen zu Eingängen. Die Aufgabe der Schaltung besteht nun darin, bei einem Schreibzugriff der CPU auf das EPROM die Programmierspannung von 25 V zuzuschalten, einen Programmierimpuls von 50 ms zu erzeugen und den WAIT-Eingang der CPU während des Anstehens des Programmierimpulses zu aktivieren, damit der Daten- und Adressenbus stabil bleiben. Nach Ablauf der 50 ms wird der WAIT-Eingang inaktiv, und die CPU beginnt mit der Abarbeitung des nächsten Befehls.

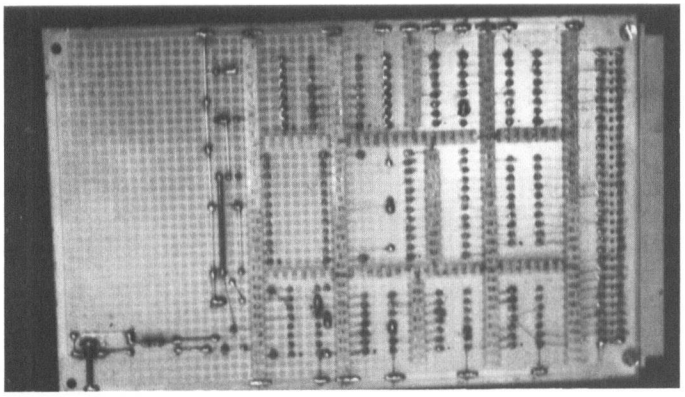
## Die Schaltungsbeschreibung

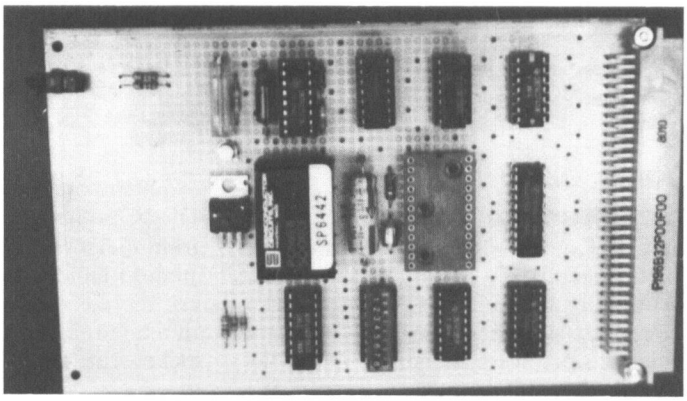
Das EPROM liegt mit den Adresseneingängen direkt und mit den Dateneingängen über den Datenbuspuffer IC2 am Bus des Z80-Systems. Der Richtungseingang von IC2 wird über die $\overline{RD}$-Leitung und der $\overline{CS}$-Eingang über die Vergleicher IC6, IC7 und einem Inverter N2 gesteuert.

## Der Lesezyklus der CPU

Startet die Z80-CPU einen Lesezyklus wird $\overline{MRQ}$ und $\overline{RD}$ nach gültig werdender Adressen- und Datenleitungen zu L-Pegel. Entspricht die Adresse auf dem Systembus der über die DIL-Schalter eingestellten Adresse am Vergleicher IC6, IC7, wird Ausgang A = B Pin 6 von IC7 zu H-Pegel und gibt die UND-Verknüpfung N6 frei.
Dadurch wird $\overline{CS}$, Pin 20 des EPROM zu L-Pegel, und das EPROM gibt seine Daten über IC2, dessen Richtungseingang ebenfalls auf L-Pegel ist, auf den Systemdatenbus, wo sie von CPU gelesen werden.







**Auf den beiden Bildern ist der mechanische Aufbau der Schaltung zu sehen. Das Bauteil mit der Aufschrift „Power Source" ist ein DC/DC-Wandler, der die +30 V zur Verfügung stellt. Für die fünf Widerstände 5,6 Ω wurde ein Widerstandsnetzwerk in DIL-Form verwendet**

Der H-Pegel von Pin 6, IC7 gelangt weiter über R1 und C1 (Differenzierstufe) und N3, N4 (Impulsformer), als Taktimpuls auf den Takteingang Pin 3 von IC4. Dieser Impuls bewirkt, daß der logische Zustand am D-Eingang Pin 2 von IC4 auf den Q-Ausgang durchgeschaltet wird. Da ein Lesezyklus stattfindet, ist $\overline{RD}$ auf L-Pegel, Ausgang Q geht somit auf L-Pegel, und Ausgang $\overline{Q}$ kippt auf H-Pegel.
Dieser H-Pegel am $\overline{Q}$-Ausgang bewirkt ein Öffnen von T1, der Adjustment-Eingang des LM317, Pin 2 wird auf L-Pegel gezogen, und am Ausgang Pin 3 des LM317 stehen ungefähr 1,2 V an.
Über die Diode D2 gelangen aber 5 V auf Pin 21 von IC1, um den Spezifikationen beim Lesen zu genügen.
Durch den L-Pegel der $\overline{RD}$-Leitung beim Lesen ist über die UND-Verknüpfung N7 der Trigger-Eingang des Monoflops IC5 gesperrt, und es werden keine WAIT-Zyklen eingefügt.
Somit verhält sich die Schaltung beim Lesen wie ein ganz normaler EPROM-Speicher und kann auch als solcher verwendet werden.

## Der Schreibzyklus der CPU

Startet die CPU einen ganz normalen Schreibzyklus in die Adresse des EPROM, so geschieht folgendes:
Die Busadresse stimmt mit der am Vergleicher IC6, IC7 eingestellten Adresse überein. $\overline{RFSH}$ ist auf H-Pegel, und $\overline{MRQ}$ wird L-Pegel. Dadurch sind alle Bedingungen erfüllt und Ausgang A = B Pin 6 von IC7 zu H-Pegel. Dieser H-Pegel gelangt auf die UND-Verknüpfung N6, kann diese aber nicht freigeben, da ein Schreibzyklus stattfindet, die $\overline{RD}$-Leitung ist somit auf H-Pegel und über N1 invertiert, die UND-Verknüpfung N6 sperrt. Der $\overline{CS}$-Eingang des EPROM bleibt auf H-Pegel.
Weiter wird der H-Pegel vom Vergleicher IC7, Pin 6 über Inverter N2 auf Pin 19 von IC2 geführt, IC2 ist also freigegeben und über eine Differenzierstufe C1, R1 und die Impulsformer N3, N4 auf den Takt-Eingang des D-Flipflop Pin 3 von IC4 und Pin 8 von N7 geführt. Die $\overline{RD}$-Leitung ist auf H-Pegel, d.h., ein H-Pegel wird auf den Q-Ausgang und ein L-Pegel auf den $\overline{Q}$-Ausgang Pin 6 von IC4 durchgeschaltet. Dieser L-Pegel sperrt T1, und am Ausgang Pin 3 des LM317 baut sich eine Spannung von +25 V auf. Diese wird über die Diode D1 auf den $U_{pp}$-Eingang des EPROM Pin 21 geführt und bereitet das EPROM auf das Programmieren vor.
Das $\overline{RD}$-Signal schaltet auch den Richtungseingang Pin 19 von IC2 von Lesen auf Schreiben und geht auf die UND-Verknüpfung N7. Ist $\overline{RD}$ auf H-Pegel (Schreibzyklus) und trifft ein Takt-Impuls an Pin 5 von N7 ein, wird das Monoflop IC5 getriggert und kippt für 50 ms in den anderen Zustand. Der $\overline{Q}$-Ausgang Pin 12 von IC5 wird zu L-Pegel, aktiviert den WAIT-Eingang der CPU, welche dann stoppt, und der Q-Ausgang Pin 5 wird für 50 ms zu H-Pegel und liefert den Programmierimpuls für das EPROM.
Während aus dem WAIT-Eingang der CPU auf L-Pegel ist, bleiben sämtliche CPU-Steuer-, Daten- und Adressenausgänge stabil, d.h., dynamische RAM, welche von der CPU aufgefrischt werden, würden unter diesen Umständen ihren Inhalt verlieren.
Nachdem das Monoflop IC5 wieder in seinen ursprünglichen Zustand zurückgekippt ist, beginnt die CPU mit der Abarbeitung des nächsten Befehls. Die +25-V-Programmierspannung bleibt bis zu einem Lesezugriff auf das EPROM stehen, was sich vorteilhaft bei Blocktransfer-Befehlen in das EPROM auswirkt, da sich die Spannung nicht vor jedem Schreibzugriff neu aufbauen muß.
Folgt nach einem Schreib- ein Lesezyklus des EPROM, passiert ein neuer Takt-Impuls an Pin 3 von IC4, die $\overline{RD}$-Leitung ist auf L-Pegel, $\overline{Q}$-Ausgang Pin 6 von IC4 wird zu H-Pegel, T1 steuert durch, und am Ausgang des LM317 sinkt die Spannung bis unter 5 V ab. Die Programmierspannung wird auf diese Weise abgeschaltet.
Jeder Schreib-(LD-)Befehl in die Adresse des EPROM, gleich ob direkt oder indirekt, führt zu einem Zuschalten der Programmierspannung und zum Erzeugen eines Programmierimpulses.
Dadurch können auch die leistungsstarken Blocktransfer-Befehle wie LDIR und LDDR zur Programmierung verwendet werden.
Nachstehend wird noch ein kleines Programm angeführt, welches die Daten, die beginnend ab der Adresse 2 „Quelldaten", in die Adresse des EPROM schreibt.
Die Anzahl der zu programmierenden Byte ist abhängig vom Wert im Register BC. Bei BC = 0800H (2048 Byte) beträgt die Programmierzeit (2048 Byte × 50 ms) etwa 100 s.

```
21 XX XX  LD HL, „Quelldaten-Adr."
11 XX XXLD DE, „EPROM-Adr."
01 XX XXLD BC, „Anzahl zu progr.
               Byte"
ED BO     LD IR
FF        RST 38 (zurück zum Moni-
               tor-Progr.)
```

Die Schaltung ist vollständig dekodiert und kann über fünf DIL-Schalter oder Drahtbrücken von 0000-F800 in 2K-Schritten über den gesamten Adressenbereich verschoben werden.
Zu beachten ist noch, daß das EPROM bei eingeschalteter Programmierspannung nicht aus seiner Fassung gezogen werden darf, da das EPROM sonst zerstört wird. Die Programmierspannung ist vorher entweder mit einem Lesebefehl oder mit einem System-Reset abzuschalten. Der Reset wirkt direkt auf den Reset-Eingang Pin 1 des D-Flipflop IC4 und setzt dieses zurück.

Peter Blaschkowitsch

# EPROM-Programmierboard

## Allgemeine Schaltungshinweise

Bei der Konzeption dieses Programmierboards stand eine möglichst universelle Anwendbarkeit auf verschiedene Mikrocomputersysteme sowie die Programmierbarkeit verschiedener EPROM-Typen im Vordergrund.

Dabei sollte der Hardwareaufwand gering bleiben und viele Funktionen möglichst auf die Software übertragen werden.

Mit dem Programmierboard lassen sich die zur Zeit verbreitetsten EPROM-Typen wie 2708, 2716 (+5 V), 2758 und 2732 programmieren bzw. lesen.

Da nun die Pinbelegung dieser EPROM teilweise voneinander abweicht, wurden die entsprechenden Signale vom PIA kommend nicht direkt, sondern über ein auswechselbares Steckmodul zum Programmiersockel geführt. Bei den Steckmoduln handelt es sich um DIL 14-Sokkel, die die notwendigen Verbindungen an die jeweiligen Pins des Programmiersockels führen.

## Die Spannungen

Als Programmierspannung werden +26 V (±1 V) für den EPROM 2708, bzw. +25 V (±1 V) für die EPROM 2716, 2758 und 2732 benötigt. Diese Spannungen liefert das Netzteil.

Die übrigen Spannungen (+5 V, −5 V, +12 V) werden einem externen Netzteil entnommen.

Für die Programmierung bzw. zum Lesen der vier EPROM-Typen sind die folgenden, umschaltbaren Spannungspegel notwendig:

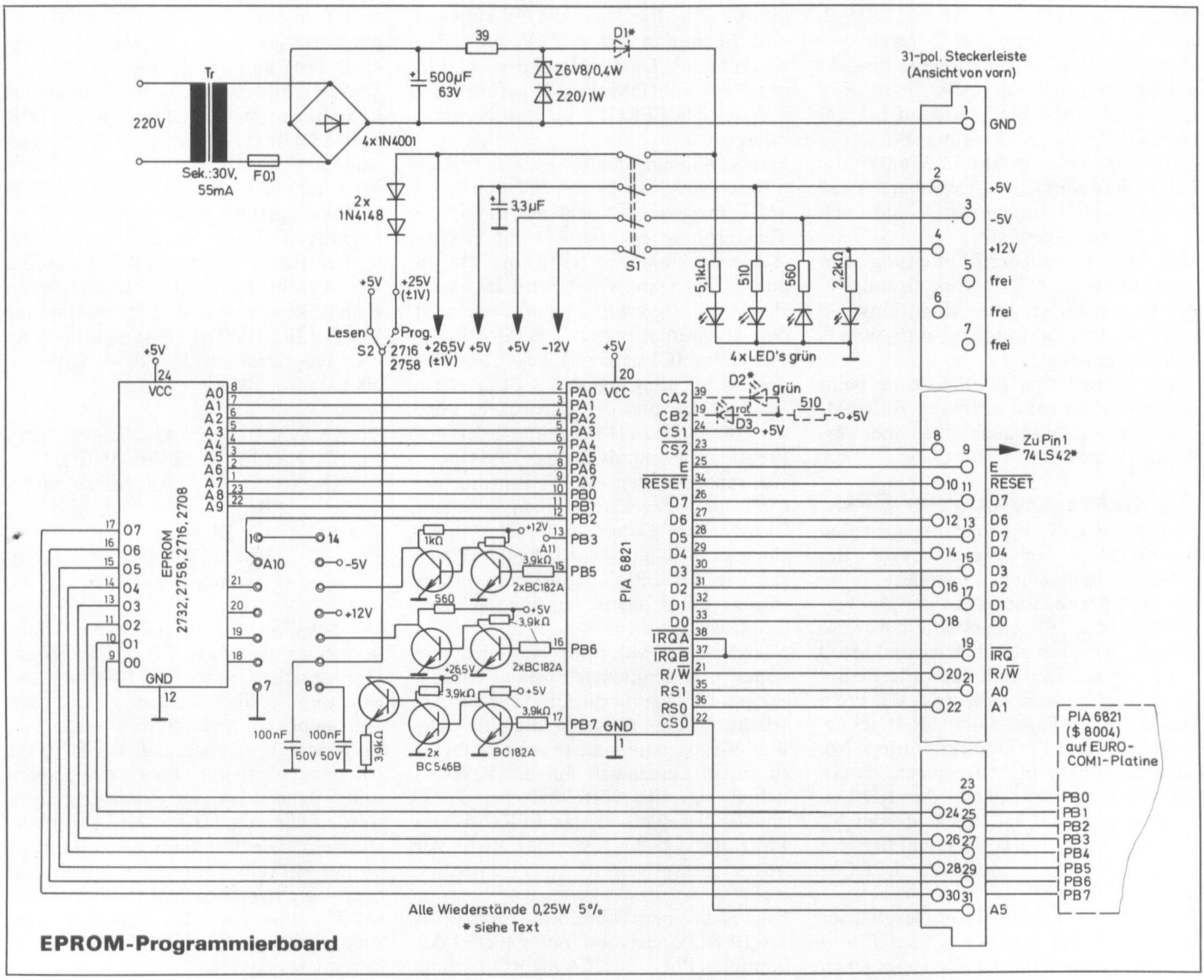
+ 5 V/GND,
+12 V/GND,
+26 V/GND,
+25 V/+5 V.

Letztere Umschaltung erfolgt nicht über ein PIA-Signal, sondern manuell mit dem Schalter S2.

Um ein Überschwingen des Programmierimpulses über die obere Grenze zu verhindern, ist jeweils am Programmieranschluß ein 100-nF-Kondensator gegen Masse geschaltet.

## Das Ein- und Ausschalten

Laut Herstellerangaben ist für die Zu- und Abschaltung der einzelnen EPROM-Spannungen eine gewisse Reihenfolge




**EPROM-Programmierboard**

einzuhalten um den EPROM nicht zu beschädigen.
Es ist auch zulässig, alle benötigten Spannungen gleichzeitig anzulegen und gleichzeitig abzuschalten, wie es mit dem Vierfach-Schalter S1 realisiert wurde.
Um eine gleichzeitige Einschaltung aller benötigten Betriebsspannungen sicherzustellen, ist der Schalter S1 erst einzuschalten, wenn durch das Leuchten der entsprechenden LED das Vorhandensein der Spannungen signalisiert wird.
Die Vorwiderstände der LED sind so dimensioniert, daß jeweils ein Strom von ca. 5 mA fließt (gleiche Helligkeit).

## Das Umschalten „Lesen" – „Programmieren"

Mit dem Schalter S2 lassen sich für die EPROM 2716 und 2758 die Betriebsarten „Lesen" und „Programmieren" einstellen. Dabei ist zu beachten, daß vor einer S2-Umschaltung der Schalter S1 ausgeschaltet werden muß, damit alle Spannungen gleichzeitig zu- bzw. abgeschaltet werden.
Bei den EPROM 2708 und 2732 ist für den Lesevorgang keine Umschaltung erforderlich, d.h., S2 bleibt auch beim Lesen in der Stellung „Prog.".

## Der Anschluß an einen Mikrocomputer mit 6800-/6802-/6809-CPU

Die Verbindung zum Mikrocomputer stellt eine 31polige Stiftleiste dar. Über diese werden folgende Signale zur Programmierplatine geführt:
- der Datenbus D0 – D7,
- die Signale E, $\overline{\text{RESET}}$, $\overline{\text{IRQ}}$ und R/$\overline{\text{W}}$,
- sowie ein freies Port (P0 – P7) eines PIA auf der Mikrocomputer-Platine.

Über diesen Port geht der Transfer der EPROM-Daten bei der Programmierung bzw. beim Lesen.
Zur Dekodierung des auf der Programmierplatine befindlichen PIA wird im einfachsten Fall ein vorhandenes Dekodiersignal von der Mikrocomputerplatine herangezogen. Als Beispiel kann die Dekodierung für den EUROCOM 1 dienen.

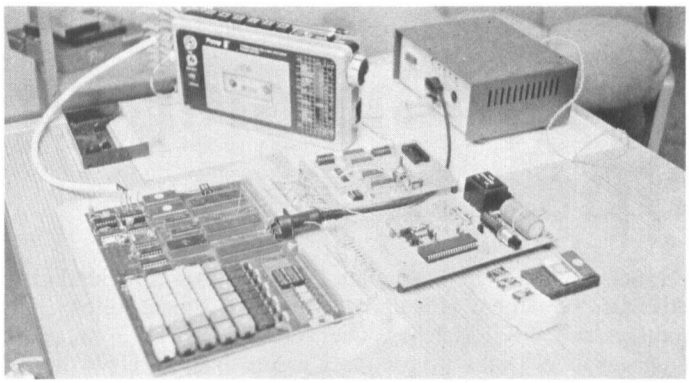
## Anschluß an EUROCOM 1

Beim Anschluß des Programmierboards an den EUROCOM 1 sind die folgenden Signale und Busleitungen erforderlich.
Vom EUROCOM 1-Board läßt sich außer den vorhandenen Spannungen +5 V, −5 V und +12 V auch der freie PIA mit der Adresse $8004 benutzen. Im Schaltungsentwurf ist Port B dieses PIA mit dem Programmierboard verbunden.
Um den auf der Programmierplatine befindlichen PIA zu adressieren, sind dessen Anschlüsse CS0 und CS1 mit H-Pegel und $\overline{\text{CS2}}$ mit L-Pegel zu belegen.
Dies wurde folgendermaßen realisiert:
- CS0 ist mit A5 des Adressenbus verbunden,
- CS1 liegt fest an +5 V, und
- $\overline{\text{CS2}}$ ist an Pin 1 des Dekoderbausteins 74LS42 auf der EUROCOM 1-Platine geführt.

$\overline{\text{CS2}}$ geht damit nur auf L-Pegel, wenn eine Adresse mit Bit 15 = 1 ($8000) auf den Bus gelegt wird.
Zusammen mit der Adressenleitung A5 sind die vier PIA-Register ab der Adresse $8020 ansprechbar.
Die vier Register haben folgende Bedeutung:

$8020 – Datenregister Port A
$8021 – Datenregister Port B
$8022 – Kontrollregister Port A
$8023 – Kontrollregister Port B

**Das Bild zeigt den EUROCOM I mit dem EPROM-Programmierboard**



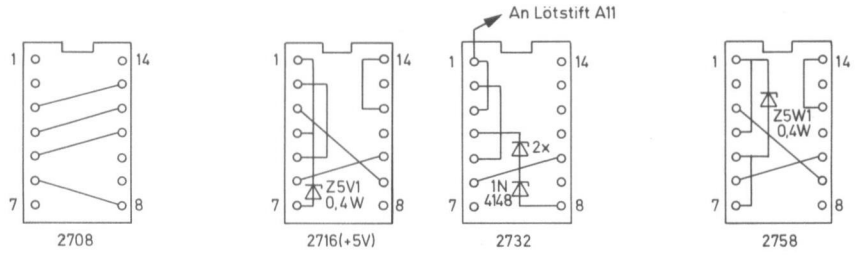
## Der praktische Aufbau des Programmierboards

Beim Aufbau empfiehlt es sich, mit der Spannungsversorgung zu beginnen und erst nach deren Prüfung den PIA anzuschließen, für den man zweckmäßigerweise eine Fassung vorsieht.
Dabei kann es durch Bauteiletoleranzen der Z-Dioden eventuell erforderlich sein, eine Si-Diode in Durchlaßrichtung einzufügen (D1), um auf die geforderte Programmierspannung von +26 V (gemessen am Steckmodul) zu kommen.
Nun können die an den PIA-Ports A und B liegenden Spannungspegel, durch Ausgabe entsprechender Signale vom Mikroprozessor, getestet werden.
Zum Schluß sind noch die Transistorstufen zur Spannungspegelumschaltung sowie der Sockel für die Steckmoduln und die EPROM-Fassung einzulöten.
Als Steckmodulsockel eignen sich vorzugsweise einzelne Sockelstifte.
Beim Abschlußtest vor der ersten Programmierung eines EPROM sind die an der EPROM-Fassung ankommenden Signale zu prüfen.

## Die Software

Für den EUROCOM 1 sind folgende Programme existent bzw. in Vorbereitung:
- PROG08, Programmierprogramm für den EPROM 2708 (2708L)
- PROG16/58, Programmierprogramm für den EPROM 2716 und 2758
- PROG32, Programmierprogramm für den EPROM 2732

Nach erfolgter Programmierung schließt sich bei jedem dieser Programme ein Prüflesen des EPROM-Inhalts an. Ergibt sich dabei eine Abweichung vom RAM-Inhalt, so wird eine entsprechende Fehlermeldung auf dem Sieben-Segment-Display ausgegeben.
Die Signalisierung des Prüfleseergebnisses ist allerdings auch über zwei LED (D2, D3 auf der Programmierplatine) möglich.




**Steckmodule für EPROM-Programmierboard**

Jürgen Beyerer

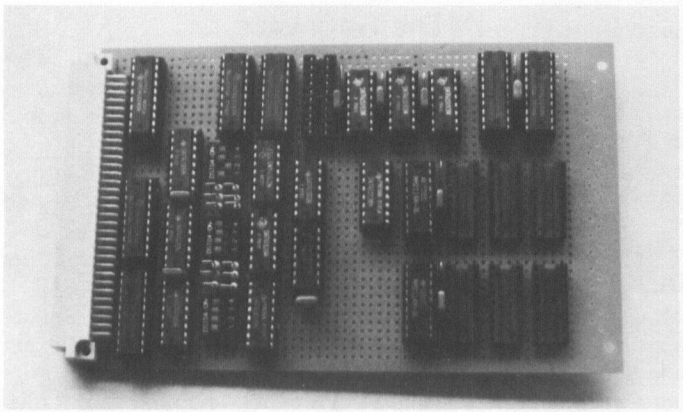
# Programmierbarer 4-KB-Stapel-Speicher

Die Schaltung arbeitet nach dem Prinzip des Stack bei einem Mikroprozessor, allerdings mit 4-KB-Speicher. Die zuletzt eingegebenen Daten werden als erstes wieder entnommen (LIFO). Das Herz des Gerätes bildet ein programmierbarer 12-bit-Vor- und Rückwärtszähler (3 × 74 LS193), der acht RAM-Bausteine des Typs 2114 adressiert. Wird die Adresse XXXF bzw. XXXE angelegt, so wird über eine Logikschaltung ein Impuls auf den Rückwärts- bzw. Vorwärtszähleingang gegeben.
Dadurch wird der an den Adressenleitungen der Speicherbausteine anliegende Zählerstand um 1 dekrementiert bzw. inkrementiert. Man kann also Datenfolgen, z.B. Tabellen, schnell abspeichern, indem man die einzelnen Werte immer wieder z.B. in Adresse XXXF speichert. Anschließend kann man die Daten durch wiederholtes Auslesen der Adresse XXXE in umgekehrter Reihenfolge auslesen. Über die Adresse XXXA können die acht niederwertigen Bit des Zählers vom Prozessor aus eingestellt werden. Über die Adresse XXXB werden die vier höchstwertigen Bit in den Zähler eingegeben. Über Adresse XXXC können die acht LSB und über die XXXD die vier MSB des Zählers ausgelesen werden. Beim Schreiben in XXXB und Lesen aus XXXD werden die oberen vier Bit des Datenbyte nicht benutzt. Die Adressenwerte XXX können an den Minidipschaltern eingestellt werden. Die zwölf Bit, die man mit den Schaltern an die eine Seite der drei Komperatoren 74LS85 legt, sind gleichzeitig die zwölf höchstwertigen Bit der vom Prozessor anzulegenden Adresse. Dabei entspricht die On-Stellung eines Schalters einer logischen Null und die Off-Stellung einer logischen Eins.

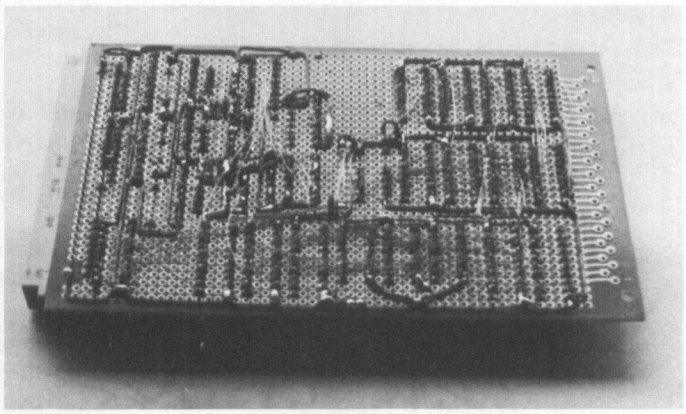
Es werden also 4-KB-Speicher zur Verfügung gestellt und dabei aber nur sechs Adressen im Adressen-Speicherraum des Computers belegt.

Das Gerät wurde an einem AIM 65 betrieben und erprobt. Die Verdrahtung wurde auf einer Lochrasterplatte mit Vero-Fädeltechnik vorgenommen.

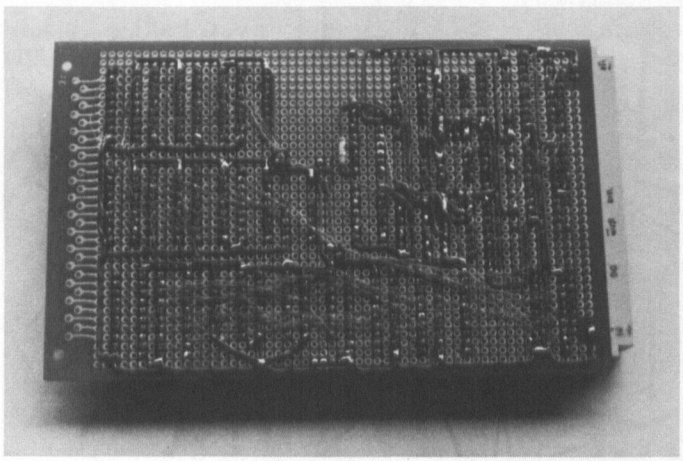
Die Bausteine sind an ihren Anschlüssen der Betriebsspannung mit je 68 nF gepuffert. Als Anschluß dient eine 64polige Steckerleiste, über die das Gerät mit dem Adressenbus, dem Datenbus und dem R/$\overline{W}$-Signal des Prozessors verbunden ist.



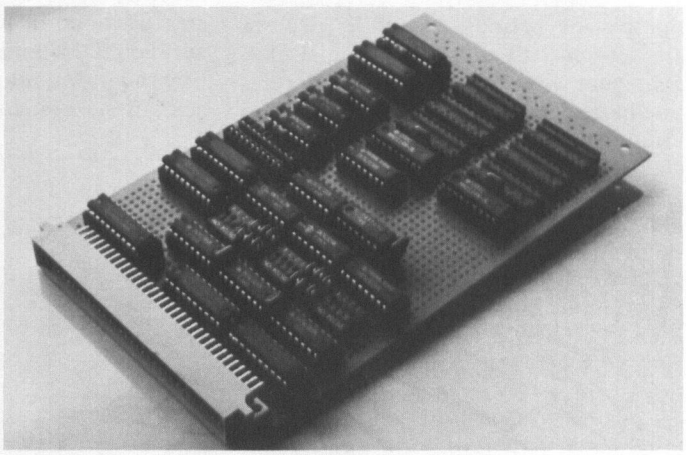
**Aufbau des programmierbaren 4-KB-Stapel-Speichers**



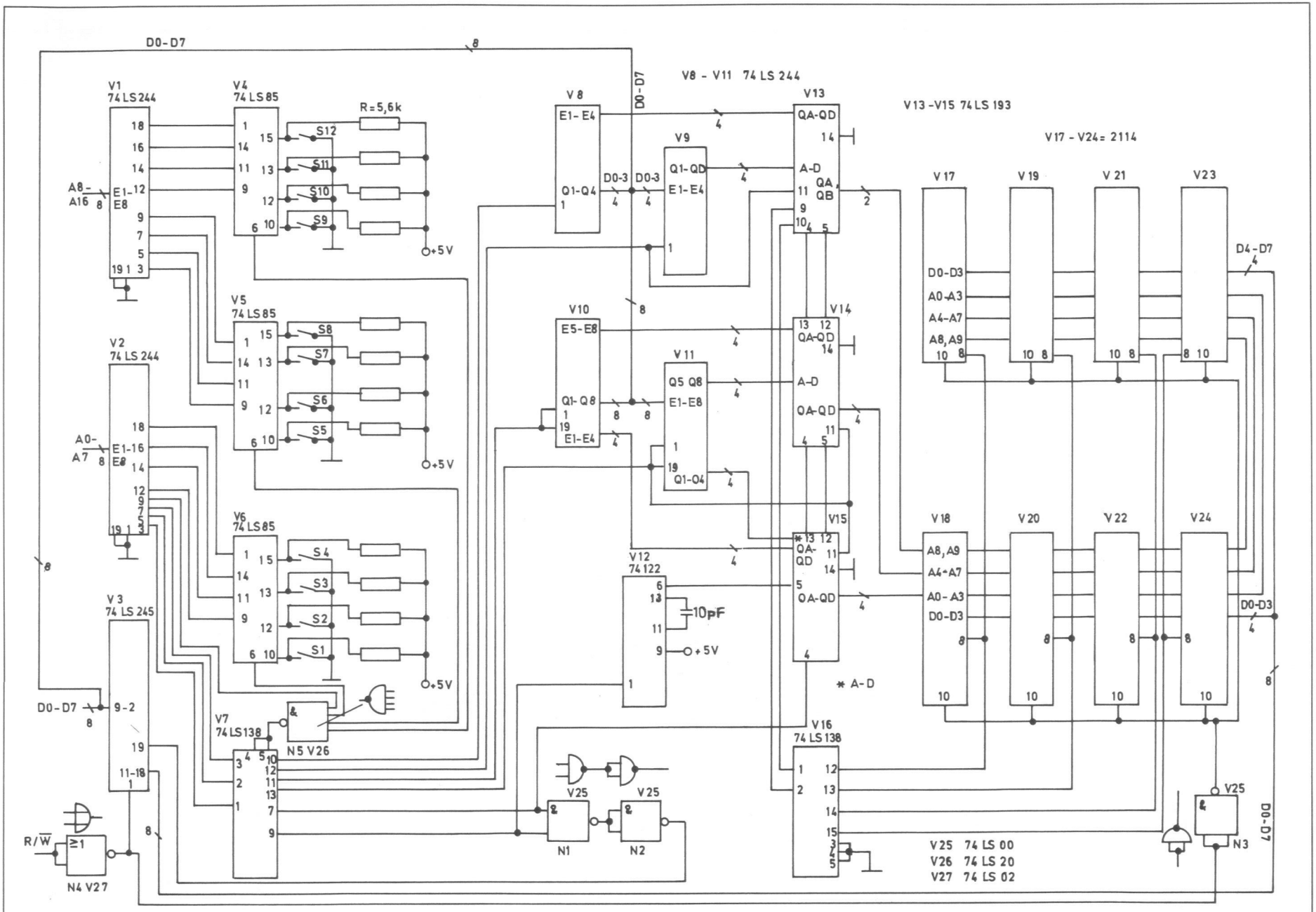
**Verbindungen an der Unterseite der Platine**



**Diese Verdrahtung macht den Entwurf einer gedruckten Platine überflüssig**



**Gesamtansicht des programmierbaren 4-KB-Stapel-Speichers mit den einzelnen Bausteinen**

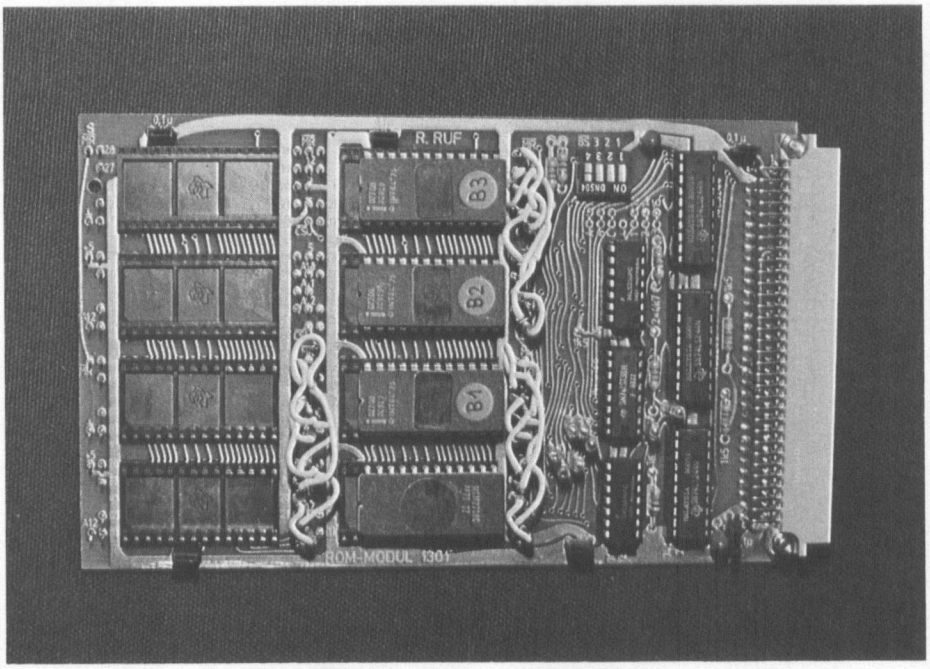



**Schaltung des programmierbaren 4-KB-Stapel-Speichers mit Standard-Schaltkreisen**

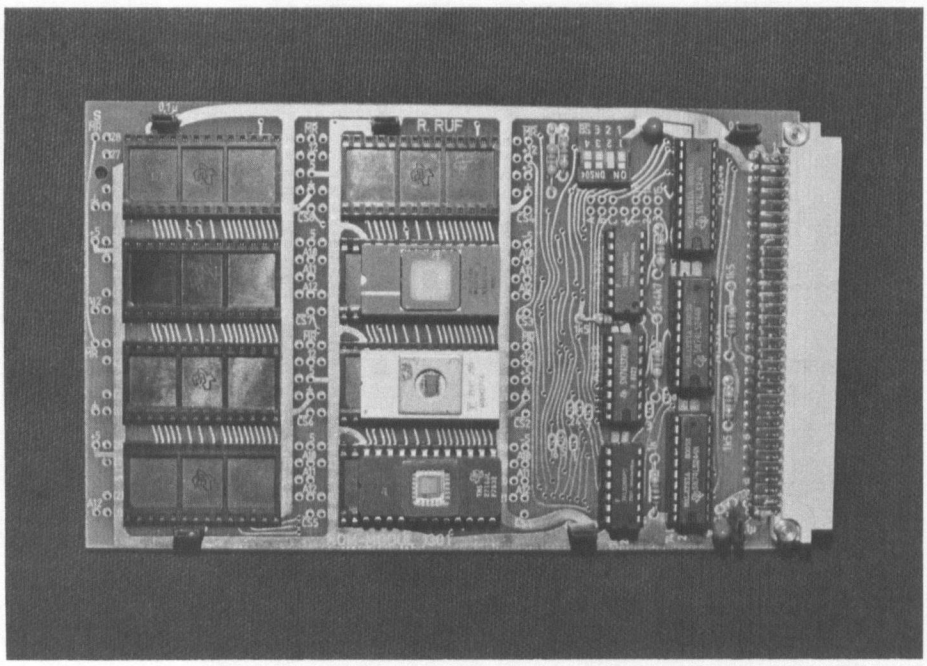
Reinhold Ruf

# ROM- bzw. EPROM-Karte für 8080/8085

In jedem Mikrocomputer ist ein Speicherbereich für das zu verarbeitende Programm enthalten. Flexibel erweiterbare Systeme werden meist modular aufgebaut, d.h., erforderliche Erweiterungen können in der Regel problemlos durch Anschließen der entsprechenden Hardware durchgeführt werden. Modular erweiterbar ist z.B. das Siemens SM-80-System. Es wurde eine preiswerte und trotzdem flexible ROM/EPROM-Erweiterungskarte entwickelt, dis buskompatibel zum SMP-80-Bus ist.




**Platinenaufbau mit Programmspeicher (EPROM)**




**Verschiedene EPROM-Typen in der Karte**

Mit acht Steckplätzen für Speicherbausteine ist die Karte von 8 KB über 16 KB und 32 KB bis 64 KB aufrüstbar. Jeder Steckplatz kann ein Speicher-IC mit der Standard-EPROM-Belegung für die Leitungen A0 bis A9, D0 bis D7, Masse und $+U_b$ aufnehmen. Die je nach Hersteller und Typ unterschiedlichen Anschlüsse (Pin 18 bis 21) sind für jeden Steckplatz den Erfordernissen des IC entsprechend frei verdrahtbar. Da die Adressendekodierung relativ einfach ausgelegt ist, kann eine Karte nur mit einer Speichergröße bestückt werden, d.h. Mindestbestückung mit z.B. 1-KB-Typen und 2-KB-Typen ist nicht möglich. Nicht bestückte Steckplätze lassen sich aus der Sicht des Mikrocomputers ausblenden.

Der Adressenbus wird mit zwei 74LS244 gepuffert. Die Leitungen A0 bis A9 des gepufferten Adressenbus liegen parallel an allen ROM-Steckplätzen. Die Datenausgänge der Speicherbausteine liegen ebenfalls parallelgeschaltet an den Eingängen der Datenbuspuffer 74LS244. Die Datenbuspuffer sind an ihren Ausgängen hochohmig, solange die Karte nicht adressiert ist.

Die Adressenbusleitungen A10 bis A15 werden je nach Kapazität der verwendeten Speicher entweder zur Adressendekodierung verwendet oder an die entsprechenden Eingänge der Speicher-IC herangeführt. Jeweils die drei niederwertigsten nicht zu ROM geführten Adressenleitungen werden mit dem Dekoder 74LS138 verbunden (Eingänge A, B und C). Über diese drei Adressenleitungen wird das Chip-Select erzeugt.

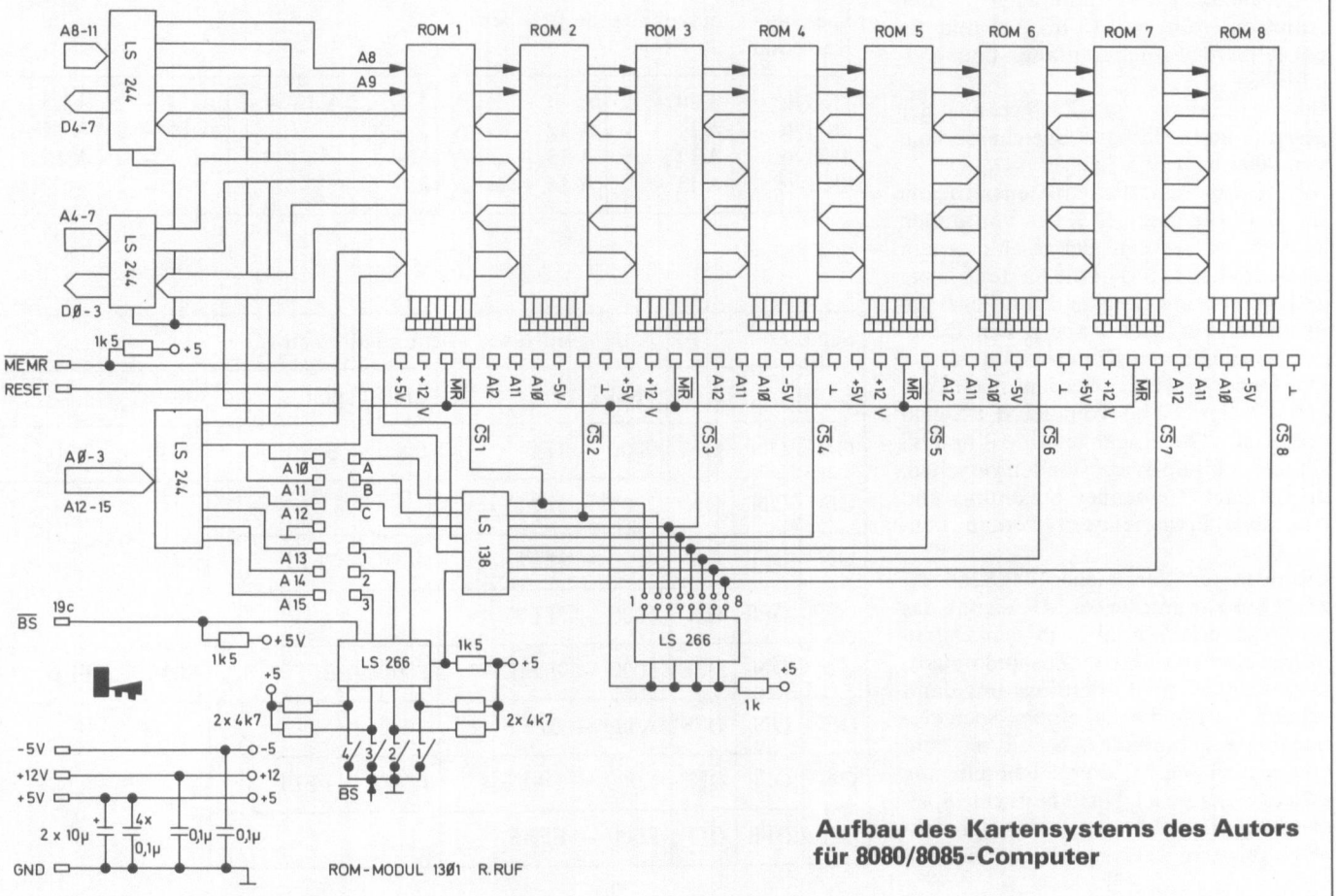
Die eventuell noch vorhandenen freien Adressenleitungen werden zur Bereichsauswahl herangezogen. Ein 74LS266 dient dabei als Vergleicher zwischen der anliegenden Adresse und der an einem DIL-Schalter eingestellten Bereichsadresse. Die freien Adressenleitungen werden dazu mit den Vergleichereingängen 1, 2 und 3 verbunden. Bei Verwendung von Speichern mit mehr als 1-KB-Kapazität bleiben u.U. mehrere Vergleichereingänge frei. Diese freien Eingänge werden mit Masse beschaltet und der zugehörige DIL-Schalter wird geschlossen. Die beiden Tabellen geben die ein-

zubauenden Verbindungen der Adressenleitungen A10 bis A15 und die einstellbaren Speicheranfangs- und -endadressen an.
Bei Verwendung von 8-KB-Typen ist der gesamte adressierbare Speicherbereich von 0000 bis FFFF belegt.
Mit einem zusätzlichen Steuereingang $\overline{BS}$ kann die gesamte Karte von außen abgeschaltet werden. Bei geschlossenem DIL-Schalter 4 ($\overline{BS}$) ist die Karte so lange gesperrt, bis am Eingang $\overline{BS}$ L-Pegel anliegt. Der Vergleicher sperrt den Dekoder bei ungültigen Kombinationen.
Der Dekoder wird außerdem noch vom $\overline{MEMR}$-Signal und vom RESET-Signal beeinflußt. Die Eingänge $\overline{MEMR}$ und $\overline{BS}$ sind mit Pull-up-Widerständen versehen, die je nach treibender Schaltung und Ausbaugrad weggelassen werden können.
Ein weiterer Vergleicher 74LS266 erzeugt ein karteninternes $\overline{MR}$-Signal, das als Speicherlesesignal auch den Datenbuspuffer in den aktiven Zustand steuert. Dieses Signal wird allerdings nur dann erzeugt, wenn die zu einem Speichersteckplatz gehörende $\overline{CS}$-Leitung zum entsprechenden Vergleichereingang durchgeschaltet ist. Nicht bestückte Speichersteckplätze können so von der Auswahl gesperrt werden, d.h., der Daten-

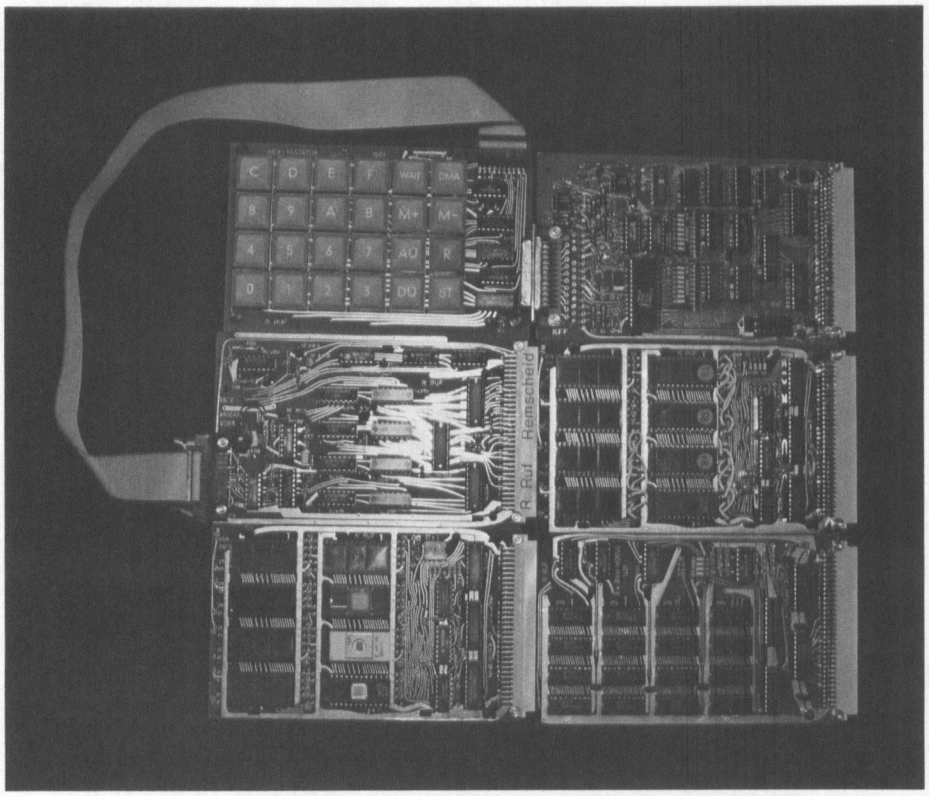
| Speicher-Kapazität | einzubauende Brücken | | | | | |
|---|---|---|---|---|---|---|
| 1KByte | A 10 – A | A 11 – B | A 12 – C | A 13 – 1 | A 14 – 2 | A 15 – 3 |
| 2KByte | A 11 – A | A 12 – B | A 13 – C | – 1 | A 14 – 2 | A 15 – 3 |
| 4KByte | A 12 – A | A 13 – B | A 14 – C | – 1 | – 2 | A 15 – 3 |
| 8KByte | A 13 – A | A 14 – B | A 15 – C | – 1 | – 2 | – 3 |

| Schalter | | | Anfangsadressen und Endadressen für | | |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 1KB-Typen | 2KB-Typen | 4KB-Typen |
| ON | ON | ON | 0000 – 1FFF | 0000 – 3FFF | 0000 – 7FFF |
| OFF | ON | ON | 2000 – 3FFF | | |
| ON | OFF | ON | 4000 – 5FFF | 4000 – 7FFF | |
| OFF | OFF | ON | 6000 – 7FFF | | |
| ON | ON | OFF | 8000 – 9FFF | 8000 – BFFF | 8000 – FFFF |
| OFF | ON | OFF | A000 – BFFF | | |
| ON | OFF | OFF | C000 – DFFF | C000 – FFFF | |
| OFF | OFF | OFF | E000 – FFFF | | |

| 2364A | 2764 | | | | | 2764 | 2364 A | 2708 | 2716 | 2332A 2732 | 2758 | 8308 2308 | 2316E | TMS 2716 | TMS 2508 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| NC | NC | | 1 | 28 | | +5 | +5 | | | | | | | | |
| A12 | A12 | | 2 | 27 | | NC | +5 | | | | | | | | |
| | A7 | 1 | 3 | 26 | 24 | +5 | | | | | | | | | |
| | A6 | 2 | 4 | 25 | 23 | A8 | | | | | | | | | |
| | A5 | 3 | 5 | 24 | 22 | A9 | | | | | | | | | |
| | A4 | 4 | 6 | 23 | 21 | A11 | A11 | −5 | +5 | A11 | +5 | −5 | +5 | −5 | |
| | A3 | 5 | 7 | 22 | 20 | $\overline{MR}$ | $\overline{MR}$ | $\overline{CS}$ | $\overline{MR}$ | $\overline{MR}$ | $\overline{MR}$ | $\overline{CS}$ | $\overline{MR}$ | A10 | $\overline{MR}$ |
| | A2 | 6 | 8 | 21 | 19 | A10 | A10 | +12 | A10 | A10 | ⊥ | +12 | A10 | +12 | NC |
| | A1 | 7 | 9 | 20 | 18 | $\overline{CS}$ | $\overline{CS}$ | ⊥ | $\overline{CS}$ | $\overline{CS}$ | $\overline{CS}$ | CS/$\overline{CS}$ | $\overline{CS}$ | $\overline{CS}$ | $\overline{CS}$ |
| | A0 | 8 | 10 | 19 | 17 | D7 | | | | | | | | | |
| | D0 | 9 | 11 | 18 | 16 | D6 | | | | | | | | | |
| | D1 | 10 | 12 | 17 | 15 | D5 | | | | | | | | | |
| | D2 | 11 | 13 | 16 | 14 | D4 | | | | | | | | | |
| | ⊥ | 12 | 14 | 15 | 13 | D3 | | | | | | | | | |




**Aufbau des Kartensystems des Autors für 8080/8085-Computer**




**Schaltung für die ROM- bzw. EPROM-Karte**

buspuffer bleibt im hochohmigen Zustand, obwohl eine gültige Adresse anliegt.

Bei Verwendung von Speicherbausteinen mit absenkbarer Verlustleistung im inaktiven Zustand (stand by) sollte die $\overline{CS}$-Leitung mit dem „Power-down"-Eingang des Speichers verbunden werden. Das $\overline{MR}$-Signal kann dann als $\overline{CS}$-Signal verwendet werden.

Durch die frei verdrahtbaren Eingänge 18 bis 21 ist es möglich, Typen mit nur einer oder mehreren Spannungen gemischt zu verwenden. Wechselt man gelegentlich die Typen, sollte man die Verdrahtung der Anschlüsse 18 bis 21 steckbar ausführen.

Der Schaltungsentwurf ist auch für die bereits lieferbaren 8-KB-Typen mit u.U. 28poligem Gehäuse vorgesehen. Beim Aufbau sind dann zusätzliche Brücken für die Anschlüsse 1 und 2 sowie 27 und 28 erforderlich, da sich die Hersteller leider nicht auf eine einheitliche Belegung einigen konnten. Die Anschlüsse 18 bis 21 der 24poligen Versionen sind bei den 28poligen Gehäusen dann die Anschlüsse 20 bis 23. Die Belegung einiger typischer Vertreter der verschiedenen ROM bzw. EPROM ist aus der Tabelle ersichtlich.

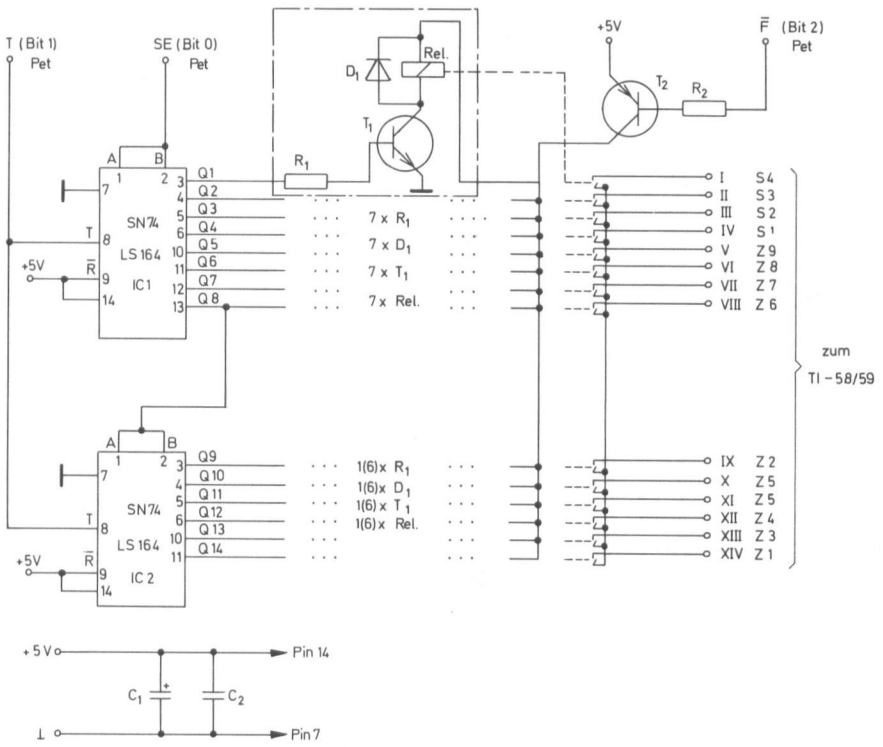
Kai Rohleff und Arno Hecker

# CBM-Interface für PC 100-Drucker

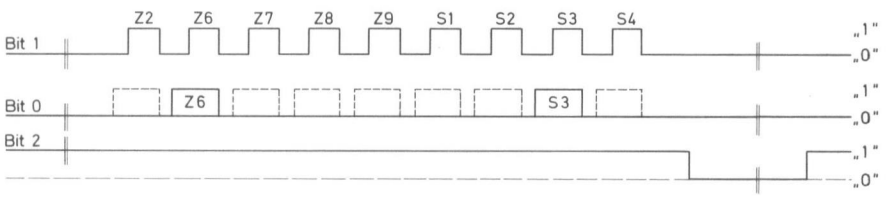
Wir entwickelten ein billiges Interface, das universell in diversen Bereichen einsetzbar ist. Zunächst zeigen wir an einem speziellen Fall, wie der Mikrocomputer PET 2001 mit Hilfe dieses Interfaces den Drucker PC 100 über die Taschenrechner TI 58/59 von Texas Instruments ansteuert:

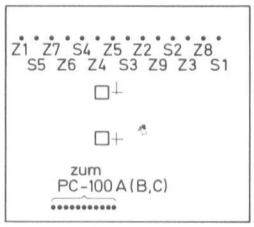
Nachdem der Anwender die zu druckende Zeile in den Bildschirmspeicher des PET eingeschrieben (es werden die ersten 20 Zeichen gedruckt; Adresse: 32768 – 32788) und das Programm mit SYS 7800 gestartet hat, wird die Zeile zunächst auf Leerzeichen untersucht. Enthält jedes Byte den Inhalt 32, so wird die Tastenfolge 2nd Adv übertragen und das Programm nach einer kurzen Pause beendet (7801 – 7832). Enthält die Zeile jedoch einen vom Leerzeichen verschiedenen Inhalt, so wird dieses Zeichen entsprechend dem Taschenrechnerkode (z.B. A = 13) entschlüsselt (7920 – 7971). Diese Zahl gibt an, welche Ziffern in den Taschenrechner übertragen werden müssen. Dazu wird die Zahl in zwei Ziffern aufgeteilt (7974 – 7990) und dann die entsprechende Spalte und Zeile übertragen (7994 – 8045). Nun wird zuerst die Information (Zeile/Spalte) auf Bit 0 ausgegeben, die dann nach wenigen Mikrosekunden auf Bit 1 getriggert, d.h. in den Speicher des Schieberegisters SN74LS164 gegeben wird. Sind alle neun Zustände übertragen, so liegt an genau zwei Ausgängen der beiden gekoppelten IC eine 1. Der zuerst übertragene Zustand befindet sich dann im zweiten Schieberegister, da jede Information bei jeder Triggerung um eine Stelle weiter verschoben wird. Nun wird die Freigabe $\overline{F}$ auf Bit 2 auf 0-Signal gesetzt. Dies bewirkt, daß die Relais an den beiden Ausgängen durchgeschaltet werden, d.h. es ist genau eine Zeile mit einer Spalte verbunden, was mit dem Drücken einer Taste zu vergleichen ist. Da der Taschenrechner zur Erfassung und Verarbeitung der Daten eine bestimmte Zeitspanne benötigt, wird die Übertragung durch eine Pause (8051 – 8075) verlängert, wobei die Dauer von der übermittelten Funktion abhängt (OP – Operationen benötigen eine längere Verarbeitungszeit als die Eingabe von Ziffern).

Nach jeweils fünf Zeichen (entspr. zehn

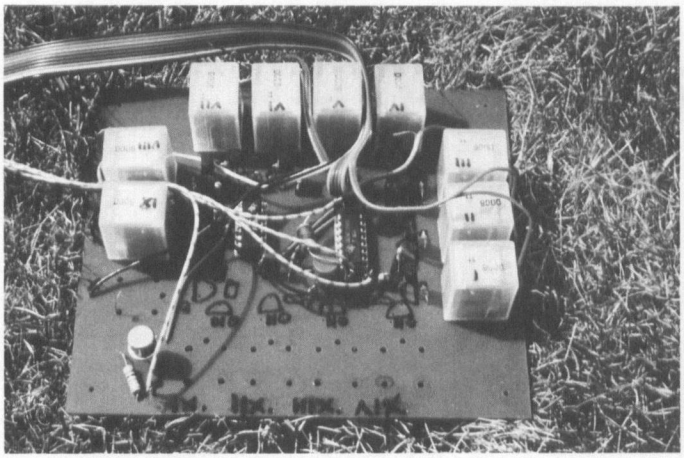



**Schaltung für ein Interface zwischen CBM-Computer und PC 100-Drucker**

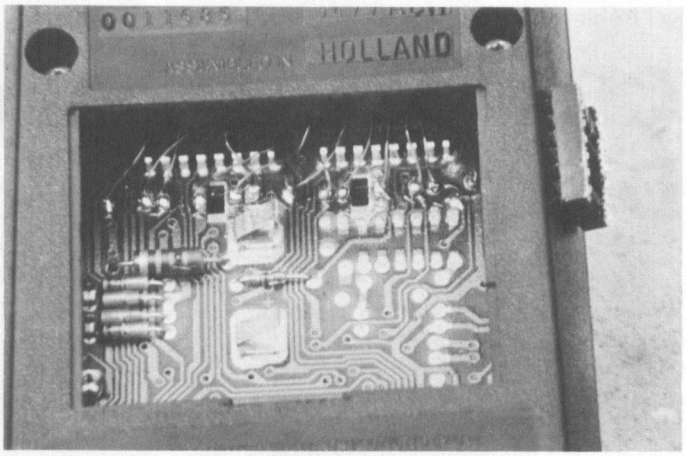


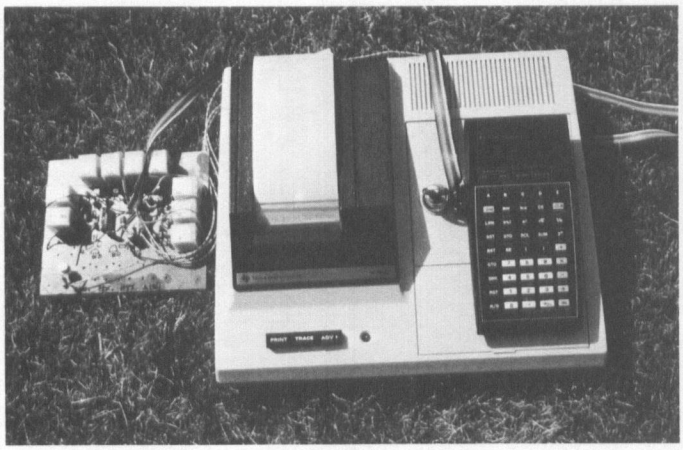

**Impulsdiagramm und Anschlußbelegung für das CBM-Interface**



**Aufbau des CBM-Interface**




**Unterseite der Platine**



**TI-Rechner und Drucker mit Interface**

**Komplette Anlage mit CBM- und TI-Geräten**

Ziffern auf dem Taschenrechner) wird die jeweilige OP-Operation übertragen, und das Programm mit dem Drucken der Zeile beendet. Die Adressen der Tabellen sind:
8080 – 8091 = Tabelle 2
8094 – 8175 = Tabelle 1

Als Beispiel zeigt Tabelle 3 die Datenübertragung zum Tasten der Ziffer 8 auf dem Taschenrechner.
Bevor das Programm gestartet wird, müssen die drei benötigten Leitungen des PET durch POKE 59459,7 als Ausgangsleitungen programmiert werden. Mit POKE 134,115 und POKE 135,30 wird das Maschinenprogramm gegen Überschreibung geschützt.
Der Aufbau der Hardware ist relativ einfach. Der Schaltplan zeigt das Interface, das zwischen dem Mikrocomputer und dem Taschenrechner geschaltet ist. Für die oben beschriebene Aufgabe benötigen wir neun Relais (≙ Grundstufe). In der Ausbaustufe benötigen wir weitere fünf Relais, um alle Zeilen und Spalten des Taschenrechners ansteuern zu können. Es ließe sich dann mit einem entsprechenden Programm eine vollkommene Steuerung des Taschenrechners realisieren, so daß man den TI 58 auch ohne Magnetkarten innerhalb von wenigen Sekunden mit seinem Programm füllen könnte.
C 1 und C 2 dienen zur Spannungsstabilisierung. Das Bild zeigt von unten gesehen die Anschlüsse vom TI 58/59. Hierbei gibt Z die Zeilennummer und S die Spaltennummer an, mit der dieser Kontakt im Taschenrechner intern verbunden ist. Um hier die Lötarbeiten leichter vornehmen zu können, entfernt man das Gehäuse, indem man die zwei Schrauben auf der Taschenrechnerunterseite löst und dann das Unterteil möglichst ohne Gewalt zu sich hin zieht. Nun wird zunächst ein 14poliger DIL-Stecker an der rechten oder linken Außenseite des Rechners angebracht, indem zuerst die Löcher für die Anschlußbeinchen gebohrt werden. Dann wird der Stecker mit einem guten Kleber oder einer Schraube am Gehäuse befestigt. Nun werden nur noch die Kontakte des Rechners mit denen des Steckers durch einen dünnen Kupferdraht verbunden.
Der Vorteil dieses Interfaces besteht in der universellen Anwendbarkeit, da sich durch die verwendeten Relais fast unbegrenzte Einsatzmöglichkeiten ergeben. (Es wird dringend davon abgeraten, die Relais durch Optokoppler für die Taschenrechneransteuerung auszutauschen. Diese funktionieren nicht, da sie im durchgeschalteten Zustand noch einen zu großen Innenwiderstand aufzeigen). Mit dieser Schaltung ließe sich beispielsweise mit nur drei Ausgangsleitungen eine Lauflichtsteuerung, ein Multifunktionstimer, eine Eisenbahnsteuerung und vieles andere realisieren. Wollen wir an jedem Relais ein Gerät anschließen, das eine bestimmte Spannung benötigt, so legen wir einfach die Leitung, die mit allen Relais verbunden ist, auf dieses Potential.
Mit POKE 59457,5 : POKE 59457,7 erhalten wir ein 1-Signal im ersten Speicher des Schieberegisters, das bei der nächsten Triggerung (Bit 1 auf 1) in den zweiten Speicher wandert.
Entsprechend mit POKE 59457,4 : POKE 59457,6 ein Low-Signal.
Durch POKE 59457,0 wird das Signal an die Relais weitergegeben. Durch POKE 59457,4 schalten die Relais wieder zurück.

```
7800: 78 A9 20 A2 15
7805: DD FF 7F D0 19
7810: CA D0 F8 A2 0A
7815: 20 3A 1F A2 0B
7820: 20 3A 1F A2 0E
7825: 20 73 1F CA D0
7830: FA 58 60 EA EA
7835: A9 00 85 00 A9
7840: 01 85 01 A9 05
7845: 85 02 A4 00 BE
7850: 00 80 AD 12 E8
7855: C9 EF F0 E5 20
7860: F0 1E 20 26 1F
7865: E6 00 A5 02 C5
7870: 00 D0 E6 A2 0A
7875: 20 3A 1F A9 7E
7880: 85 06 A2 09 20
7885: 3A 1F A2 00 20
7890: 3A 1F A6 01 20
7895: 3A 1F E6 01 18
7900: A5 02 69 05 85
7905: 02 C9 19 90 C1
7910: F0 D9 A2 0F B0
7915: A5 EA EA EA EA
7920: E0 01 90 0A E0
7925: 1B B0 06 BD 9D
```

```
7930: 1F 60 EA EA E0
7935: 27 90 0A E0 3B
7940: B0 06 BD 91 1F
7945: 60 EA EA 8A A0
7950: 12 D9 CB 1F F0
7955: 08 88 D0 F8 A9
7960: 28 60 EA EA 98
7965: 69 11 A8 B9 CB
7970: 1F 60 EA EA A2
7975: 00 C9 0A 90 05
7980: E9 0A E8 D0 F7
7985: 48 20 3A 1F 68
7990: AA EA EA EA A0
7995: 09 A9 00 E0 0A
8000: D0 0E A9 01 D0
8005: 0A EA A9 04 3E
8010: 90 1F 90 02 A9
8015: 05 8D 41 E8 49
8020: 02 8D 41 E8 A9
8025: 04 8D 41 E8 88
8030: D0 E7 3E 90 1F
8035: A9 00 20 70 1F
8040: A9 04 20 70 1F
8045: 60 EA EA 8D 41
8050: E8 A9 78 24 06
8055: 70 0D A8 88 D0
```

```
8060: FD E9 01 D0 F8
8065: 60 EA EA EA EA
8070: 06 06 A9 B4 70
8075: ED EA EA EA EA
8080: 14 24 22 21 44
8085: 42 41 84 82 81
8090: 08 12 00 00 0D
8095: 0E 0F 10 11 15
8100: 16 17 18 19 1A
8105: 1B 1E 1F 20 21
8110: 22 23 24 25 29
8115: 2A 2B 2C 2D 2E
8120: 41 37 38 33 2F
8125: 39 14 28 3F 01
8130: 02 03 04 05 06
8135: 07 0A 0B 0C 3E
8140: 20 3D 1E 21 5E
8145: 7A 25 3F 22 23
8150: 24 3B 3C 3E D1
8155: 91 DD 9D 00 40
8160: 3C 49 35 34 3D
8165: 47 41 4B 4D 4C
8170: 4A 43 3C 46 32
8175: 36 24 24 24 24
```

| Zeichen | TI-Code | PET-Code | Zeichen | TI-Code | PET-Code | neues (PET-) Zeichen |
|---|---|---|---|---|---|---|
| A | 13 | 1 | . | 40 | 46 | |
| B | 14 | 2 | / | 63 | 47 | |
| C | 15 | 3 | 0 | O1 | 48 | |
| D | 16 | 4 | 1 | 02 | 49 | |
| E | 17 | 5 | 2 | 03 | 50 | |
| F | 21 | 6 | 3 | 04 | 51 | |
| G | 22 | 7 | 4 | 05 | 52 | |
| H | 23 | 8 | 5 | 06 | 53 | |
| I | 24 | 9 | 6 | 07 | 54 | |
| J | 25 | 10 | 7 | 10 | 55 | |
| K | 26 | 11 | 8 | 11 | 56 | |
| L | 27 | 12 | 9 | 12 | 57 | |
| M | 30 | 13 | : | 62 | 58 | |
| N | 31 | 14 | | 00 | 32 | |
| O | 32 | 15 | = | 64 | 61 | |
| P | 33 | 16 | ↑ | 60 | 30 | |
| Q | 34 | 17 | ! | 73 | 33 | |
| R | 35 | 18 | π | 53 | 94 | |
| S | 36 | 19 | √ | 52 | 122 | |
| T | 37 | 20 | % | 61 | 37 | |
| U | 41 | 21 | ? | 71 | 63 | |
| V | 42 | 22 | ' | 65 | 34 | " |
| W | 43 | 23 | Δ | 75 | 35 | # |
| X | 44 | 24 | Σ | 77 | 36 | $ |
| Y | 45 | 25 | Π | 76 | 38 | & |
| Z | 46 | 26 | II | 74 | 60 | < |
| ( | 55 | 40 | $\bar{x}$ | 67 | 62 | > |
| ) | 56 | 41 | ↑ | 60 | 209 | Cursor ↑ |
| * | 51 | 42 | $^2$ | 70 | 145 | Cursor ↓ |
| + | 47 | 43 | $^x$ | 50 | 221 | Cursor ← |
| , | 57 | 44 | $^e$ | 54 | 157 | Cursor → |
| - | 20 | 45 | | | | |

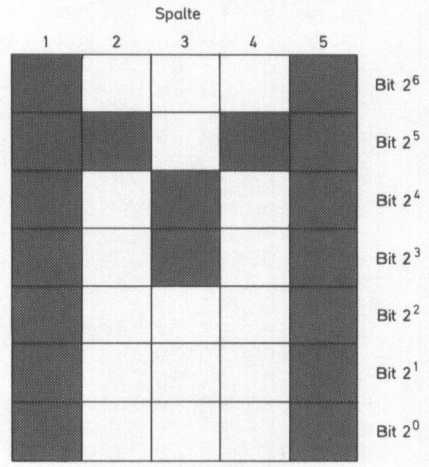
| Taste | serielle Information Z2 | Z6 | Z7 | Z8 | Z9 | S1 | S2 | S3 | S4 | dezimal ohne Z2 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 20 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 36 |
| 2 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 34 |
| 3 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 33 |
| 4 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 68 |
| 5 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 66 |
| 6 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 65 |
| 7 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 132 |
| 8 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 130 |
| 9/OP | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 129 |
| 2nd | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 8 |
| ./Adv | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 18 |

Z = Zeile
S = Spalte

Horst Fiëtta

# Interface für Drucker

Für einen Mikrocomputer ist ein Drukker ein sehr nützliches Hilfsmittel, um z.B. Programme zu dokumentieren oder einen Ausdruck der Rechenergebnisse zu erhalten. Ein Drucker der Mittelklasse ist der Mosaikdrucker 115 DR von Valvo. Er ist ein Mosaikdrucker, d.h., er schreibt Zeichen mit Hilfe einer Matrix aus Punkten. Pro Zeile schreibt er maximal sieben Punkte übereinander und 240 Spalten über die Papierbreite von 11 cm. Er kann also 40 Zeichen aus einer 5 × 7-Punkt-Matrix pro Zeile schreiben. Sein Konzept läßt aber auch das Schreiben von größeren Buchstaben und sogar Plotten zu.



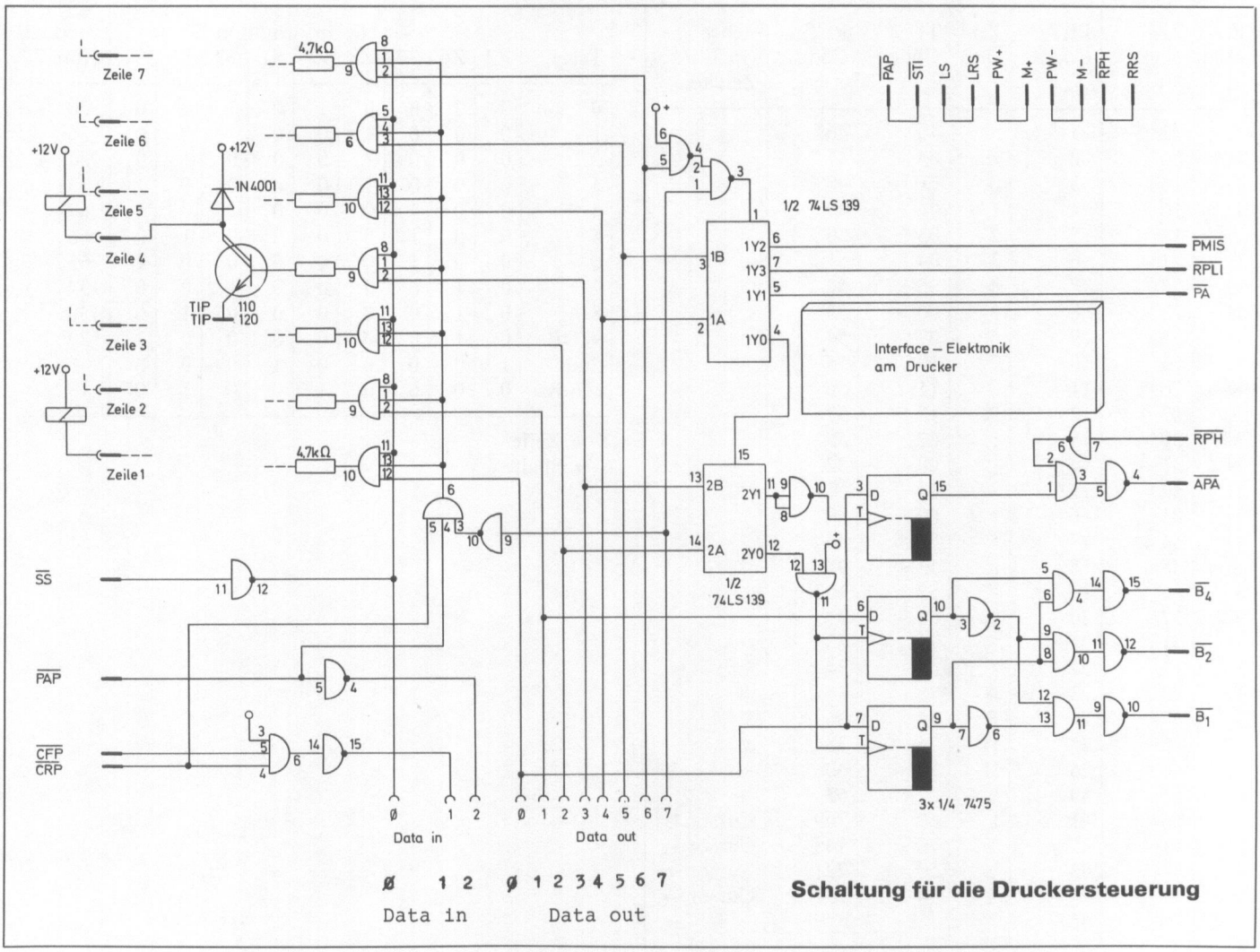

**Aufbau der Drucksteuerung**

## Das Interface

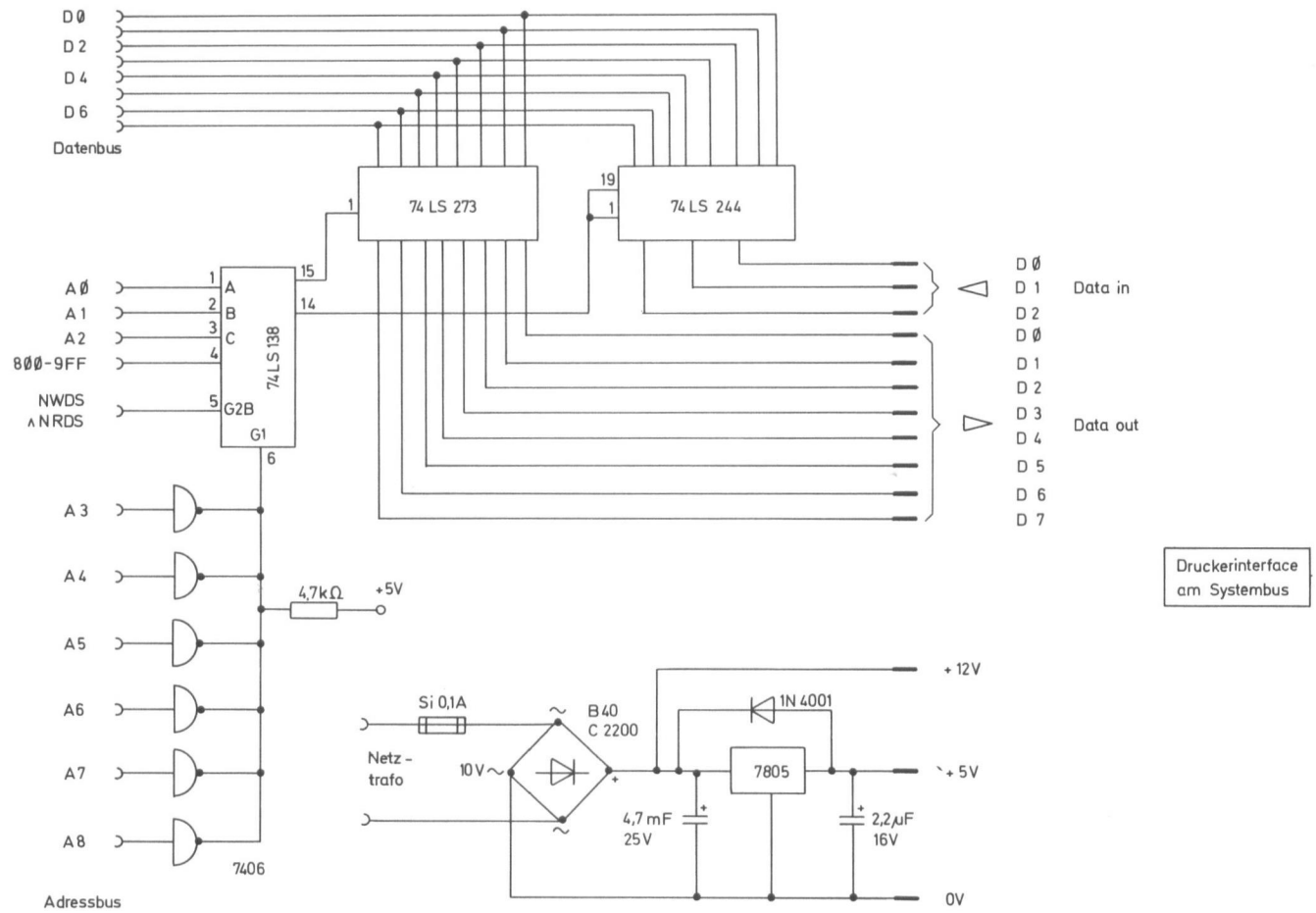
Der Drucker hat als Grundausstattung eine Minimalelektronik, die es jedoch nicht gestattet, den Drucker einfach an ein Mikroprozessorsystem anzuschließen. Das Bindeglied ist ein Interface. Es wurde darauf geachtet, daß das Interface möglichst wenig Verbindungsleitungen mit dem Computer hat, um Störungen zu vermeiden. Es ermöglicht dem Drucker außerdem, alle vorhandenen Möglichkeiten voll zu nutzen. Um das Interface an fast allen Prozessorsystemen betreiben zu können, wurde es in zwei Teile aufgeteilt. Der erste Teil liegt direkt am Systembus und hat acht Outputleitungen und drei Inputleitungen, die zum zweiten Teil des Interface führen. Der zweite Teil ist im Gehäuse des Druckers untergebracht und übergibt die Signale der Minimalelektronik.

## Die Voreinstellungen

Um mit dem Drucker richtig arbeiten zu können, müssen erst einige Signale (Voreinstellungen) erzeugt werden. Der 115 DR läßt es dem Benutzer offen, ob nach dem Zeilenende der Papiervorschub automatisch oder manuell geschieht. Der




**Schaltung für die Druckersteuerung**

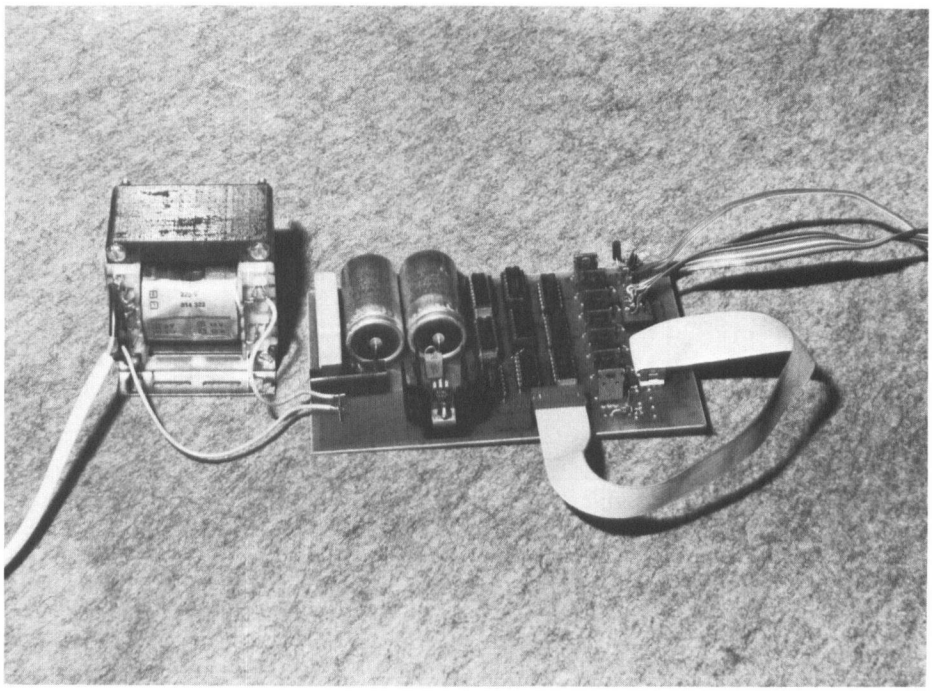



**Interfaceschaltung für die Druckersteuerung**

Computer vermittelt zunächst einen Befehl „Automatischer Papiervorschub ein/aus". Als nächsten Befehl erhält der Drucker die Anweisung, wie groß der Papiervorschub gewählt wurde. Es gibt vier Möglichkeiten: 1,47/2,94/4,41/5,88 mm. Hierfür wurden vier Befehle ausgewählt. Die Informationen über den Papiervorschub und -automatik bleiben im Interface gespeichert, bis sie durch eine andere Information überschrieben werden.

## Der Druckbeginn

Für den Start des Druckvorganges benötigt der Drucker einen Befehl. Dieser kommt ebenfalls vom Computer, wird aber nicht im Interface gespeichert. Nach dem Druckerstart werden nun die Informationen für die sieben Schreibspulen (jede Schreibspule erzeugt einen Punkt) über die Datenleitungen zum Interface gebracht. Wird vor Ende der Zeile ein Zeilenrücklauf gewünscht, so kann dies ebenfalls durch einen Befehl geschehen.



**Aufbau für das Drucker-Interface**

**Ein Beispiel für die 5 × 7-Matrix**
Buchstabe „M":
Dateninformation:

| | Hex | Bit | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
|---|---|---|---|---|---|---|---|---|---|
| Spalte 1 | 7F | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Spalte 2 | 20 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Spalte 3 | 18 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| Spalte 4 | 20 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Spalte 5 | 7F | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Die Liste der Befehle:**

| Hex Befehl | Steuerung | Signal |
|---|---|---|
| A0 | Druckerstart | $\overline{PMIS}$ |
| B0 | Schlittenrücklauf aus jeder Position | $\overline{RPH}$ |
| 90 | Papiervorschub | $\overline{PA}$ |
| 85 | Automatischer Papiervorschub ein | $\overline{RPH} \rightarrow A\overline{PA}$ |
| 84 | Automatischer Papiervorschub aus | |
| 80 | Papiervorschublänge: 1,47 mm | $\overline{B1}$ |
| 81 | Papiervorschublänge: 2,94 mm | $\overline{B2}$ |
| 82 | Papiervorschublänge: 4,41 mm | |
| 83 | Papiervorschublänge: 5,88 mm | $\overline{B4}$ |

**Das Druckerprogramm**

```
0E00  C4 0C 35 C4 00 31 C4 08 36 C4 00 32 C4 A0 CA 00
0E10  C2 00 D4 01 9C 08 C2 00 D4 02 98 14 90 F2 C5 01
0E20  CA 00 C2 00 D4 01 98 02 90 F8 C4 00 CA 00 90 E0
0E30  00 2A 48 4F 52 53 54 20 46 49 45 54 54 41 2A 00
```

Obiges Listing ist ein Auszug aus dem Druckprogramm (für normalen Zeichendruck). Es ist für das SC/MP-System und schreibt eine übersetzte Textzeile (ab Adr. 0C00) in das Druckerinterface (Adr. 0800).

Sollte die vorgeschobene Papierlänge nicht ausreichen, kann ein weiterer Befehl mehr Papier transportieren.

## Die Realisierung des Interface

Das Interface kann als ein Speicherzellenpaar angesehen werden, besitzt also auch nur eine Input- und eine Output-Adresse. In dieser Speicheradresse werden nun die Befehle und die Informationen für die Schreibspulen geschrieben. Die Adressenauswahl geschieht mit dem Adressendekoder, der die Inputlatches und die Outputdriver initialisiert. Die Dateninformation dient nur zur Erzeugung der Druckerbefehle. Das Interface entscheidet mit Hilfe von Bit $2^7$, ob es sich um einen Befehl oder um Druckinformation handelt. Befehle werden mit dem Dekoder entschlüsselt und ergeben entsprechende Signale für den Drucker. Die Auswahl des Papiervorschubs wird mit den Bit $2^0$ bis $2^3$ vorgenommen. Diese Information wird aber in den Latches zwischengespeichert, da sie ständig als Signal am Druckereingang liegen muß. Werden die Schreibspulen initialisiert, so tritt die Befehlsdekodierung außer Kraft und die Dateninformation gelangt an die UND-Gatter mit drei Eingängen. Ein Schreibspulensynchronsignal schaltet die Daten zu den Schreibspulen durch. Es wird damit verhindert, daß sich die Schreibspulen zu stark erwärmen. Vom Interface gelangen noch einige nützliche Signale über die Outputdriver auf den Datenbus. Zur Synchronisation des Prozessors liegt das Schreibspulensynchronsignal an Bit $2^0$. An Bit $2^1$ ein Signal, das während der Schreibzeit auf H-Pegel ist, und an Bit $2^2$ lassen sich die Papiervorschubeinheiten zählen.

## Die Software zur Steuerung

Das Programm zum Betrieb des Druckers ist in Maschinensprache abgefaßt. Es läßt sich bei Bedarf auch von einer höheren Programmiersprache aufrufen, so daß der Drucker als Dokumentationshilfe eingesetzt werden kann. Das Programm muß zunächst die in einem String abgelegten ASCII-Zeichenkodes in für den Drucker verarbeitbare Information übersetzen. Der Drucker kann dabei maximal 40 Zeichen verarbeiten. Eine übersetzte Textzeile besteht aus der Information für die 240 Spalten. Jedes Datenbit, außer Bit $2^7$, kann jeweils einen Punkt setzen. Bit $2^0$ setzt z.B. den untersten Punkt in einer Spalte, Bit $2^6$ den obersten. Aus dieser Punktmatrix läßt sich jede beliebige Zeichenform herstellen.
Es läßt sich leicht einsehen, daß sich auch größere Zeichen realisieren lassen, wenn man nach Zeilenende den Papiervorschub so eingestellt hat, daß sich die nächste Zeile nahtlos an die erste fügt. In gleicher Weise kann auch der Faksimiledruck (Plotten) erstellt werden. Ist die Information für eine Zeile übersetzt und abgespeichert, so müssen als nächstes die Voreinstellungen am Drucker vorgenommen werden. Danach kommt das Druckerstartsignal, und der Computer wartet, bis das Schreibspulensynchronsignal kommt. Dann gibt er die erste Information auf den Datenbus und wartet, bis das Signal wieder verschwunden ist. So schreibt der Drucker 240 Spalten oder weniger. Falls das Zeilenendesignal vorher kommt, springt der Computer aus dem Druckprogramm.

Reinold Ruf

# HEX-Tastatur für ein Mikroprozessor-Modulsystem

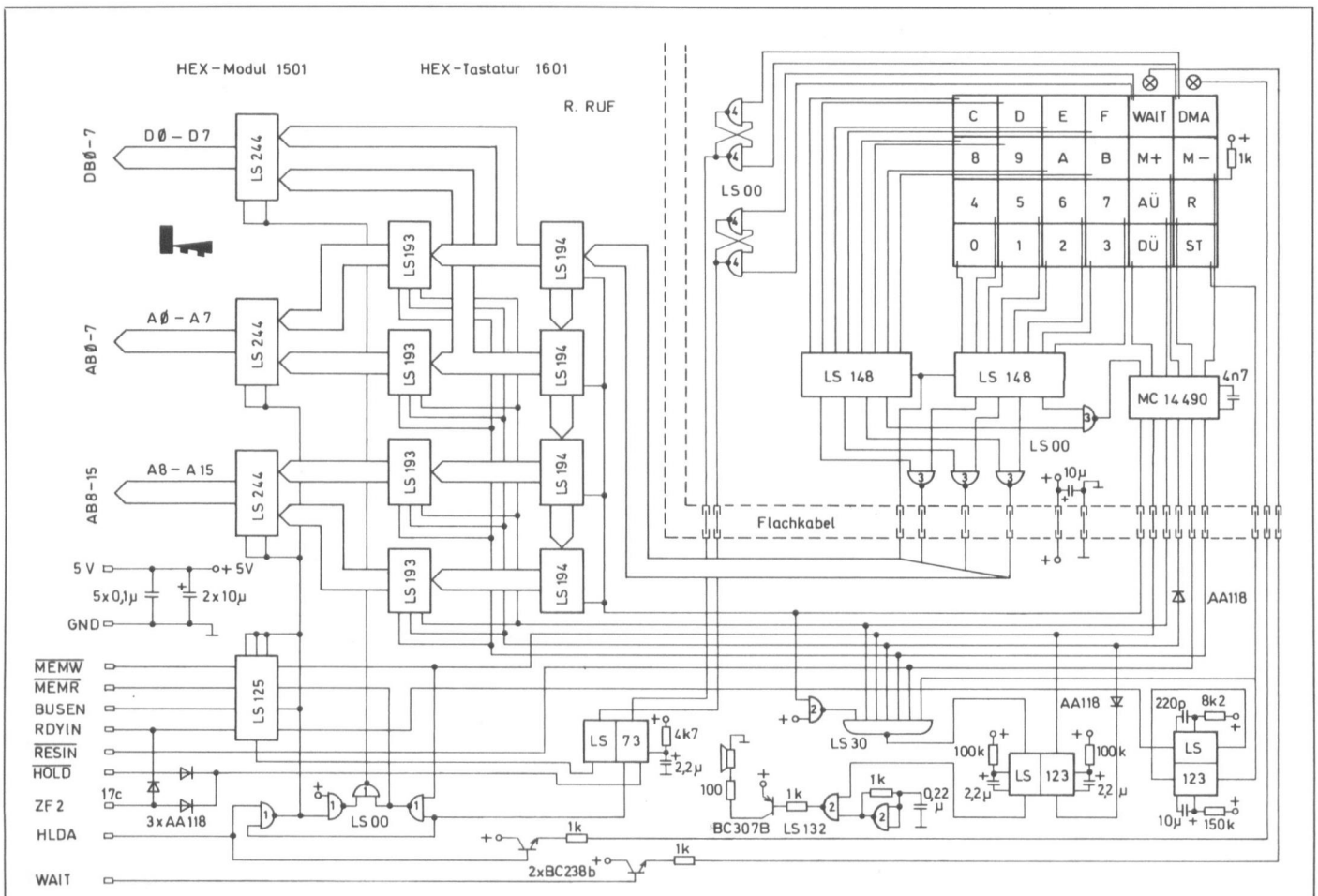
An vielen Mikrocomputern ist zur Mensch-Maschine-Kommunikation eine Tastatur angeschlossen. In einfachen Systemen, besonders in Lernsystemen, reicht es häufig aus, eine HEX-Tastatur zu verwenden. Damit wäre die Möglichkeit gegeben, mit dem Mikrocomputer auf Maschinensprachebene zu arbeiten. Die Aktivitäten des Mikrocomputers können dabei auf einer 7-Segment-Anzeige erfolgen. Die Tastatur bietet noch einige Steuerfunktionen, die nachfolgend beschrieben werden.

Die meisten HEX-Tastaturen arbeiten softwaregesteuert über eine parallele Schnittstelle (meist 8255). Die hier beschriebene Schaltung verzichtet auf Software und arbeitet direkt am Systembus per DMA. Mit einer solchen Lösung kann sehr anschaulich in die Mikroprozessortechnik eingestiegen werden.

Das Tastenfeld der HEX-Tastatur besteht im wesentlichen aus den 16 Tasten 0 bis F, die über einen HEX-Kodierer aus zwei 74LS148 und einem 74LS00 in HEX-Zahlen umgesetzt werden. Jeder Tastendruck wird über den Strobeausgang des Kodierers und eine nachgeschaltete Entprellschaltung entprellt. Nach der Entprellzeit wird das gültige HEX-Zeichen auf das erste Glied einer Schieberegisterkette aus vier 74LS194 weitergegeben. Das oberste Schieberegister übernimmt das anliegende, gültige HEX-Zeichen. Wird eine weitere HEX-Taste gedrückt, wird das erste Zeichen in das zweite Schieberegister übergeben, und das neu hinzugekommene Zeichen geht in das erste Schieberegister. So lassen sich bis zu vier HEX-Zahlen eingeben, wobei die zuerst eingegebene Zahl an der höchstwertigen Stelle steht. Jedes weitere hinzukommende Zeichen schiebt das im höchstwertigen Schieberegister stehende Zeichen hinaus, d.h., das zuerst eingegebene Zeichen ist verloren.

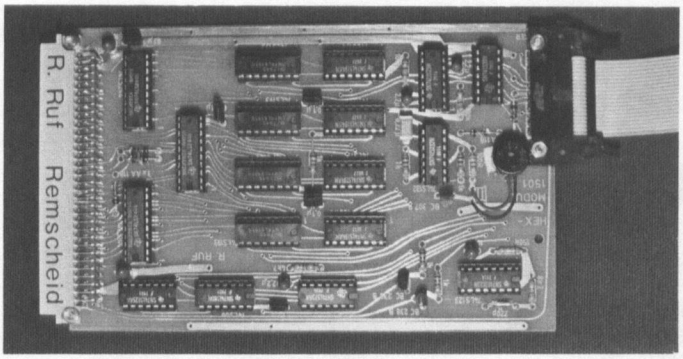
Damit die eingegebenen letzten vier HEX-Zeichen als Adresse oder die letzten beiden eingegebenen Zeichen als Datenbyte an den Systembus gegeben werden können, sind folgende Steuerungen erforderlich:

Der Mikroprozessor muß in den HOLD-Zustand (DMA) versetzt werden. Dies geschieht durch Anlegen eines L-Pegels an die HOLD-Leitung. Dazu wird die DMA-Taste einmal betätigt. Über eine

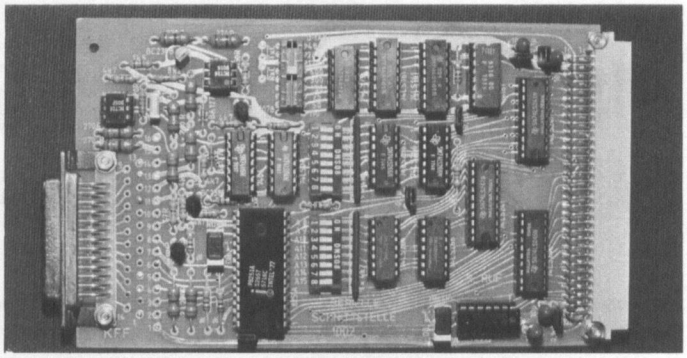



**Schaltung für eine HEX-Tastatur**

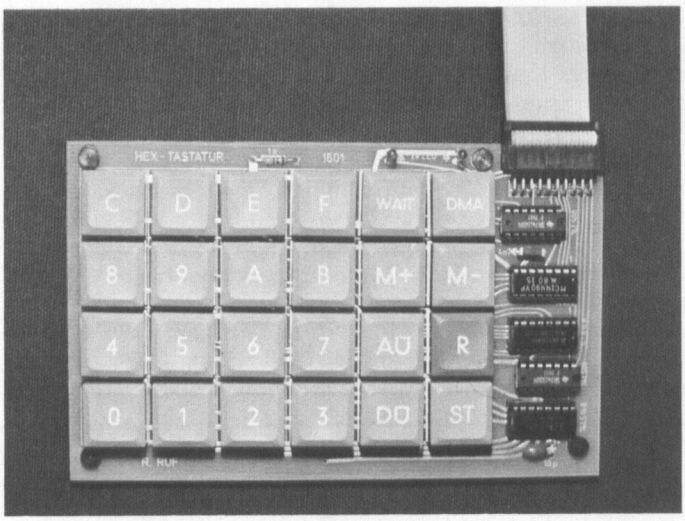
# HEX-Tastatur für ein Mikroprozessor-Modulsystem




**Aufbau der Platine für die HEX-Tastatur**



**Platinenaufbau für die serielle Schnittstelle**




**Anordnung und Aufbau einer HEX-Tastatur für ein Mikroprozessor-Modulsystem**

Entprellschaltung aus zwei NAND-Gattern wird ein Flipflop gesetzt, das den erforderlichen L-Pegel auf die HOLD-Leitung gibt. Der Mikroprozessor beendet daraufhin den laufenden Zyklus und geht in den HOLD-Zustand. Er gibt den Systembus frei und schaltet seine Ausgänge hochohmig. Über den Ausgang BUSEN wird der Datenbusausgang der Systemsteuerschaltung auf der Prozessorkarte ebenfalls hochohmig.

Über die HLDA-Leitung signalisiert der Mikroprozessor den HOLD-Zustand. Diese Quittung wird über einen Transistorverstärker an einer LED angezeigt. Durch Betätigen der AÜ-Taste (Adressenübernahme) wird die in den vier Schieberegistern stehende 16-bit-Adresse an den Synchronzähler aus vier 74LS193 parallel übergeben. Die Adresse steht dann über den Adressenpuffern 74LS244 am Adressenbus. Die anstehende Adresse kann jederzeit geändert werden, indem die neue Adresse eingegeben und die AÜ-Taste betätigt werden.

Ist ein gültiges Datenbyte in die ersten beiden Schieberegister eingegeben, kann durch Betätigen der DÜ-Taste (Datenübernahme) das Datenbyte über den Datenbuspuffer 74LS244 an den Datenbus gegeben werden. Gleichzeitig wird über den Tri-State-Steuerbuspuffer 74LS125 das $\overline{\text{MEMW}}$-Signal gegeben. Der adressierte Speicher übernimmt das angebotene Byte.

Die Übernahme eines Datenbyte ist allerdings nur möglich, wenn die Logikschaltung aus vier NAND-Gattern (1) den Steuerbuspuffer in den aktiven Zustand geschaltet hat. Dazu wird die HLDA-Quittung mit der HOLD-Anforderung der Karte verknüpft. Hat ein anderes Gerät den HOLD-Zustand verursacht, bleibt die Tastatur über diese Verknüpfung gesperrt.

Nach dem Loslassen der DÜ-Taste wird über ein Monoflop verzögert ein Zählimpuls an den Synchronzähler gegeben. Dieser Zählimpuls geht auf den Vorwärtseingang des Zählers und erhöht die dort stehende Adresse um 1, die nächste Adresse steht am Adressenbus und in diese kann das nächste Datenbyte übergeben werden.

Will man den Erfolg der so vorgenommenen Programmierung kontrollieren, kann mit der M+-oder M--Taste der Adressenzähler um jeweils einen Schritt nach oben oder unten gezählt werden. Damit das zu Anfang erwähnte Anzeigemodul die Daten und Adressen auch anzeigt, ist die Verdrahtung der Leitung ZF 2 erforderlich. Diese aktiviert den Strobeeingang des Anzeigemoduls und die von der Tastatur angebotenen Daten und Adressen werden ständig angezeigt.

Durch zusätzliche Hardware ist mit der Tastatur auch Einzelschrittbetrieb möglich. Dazu wird der Mikroprozessor in Wartezyklen (WAIT) gesteuert. Nach dem Betätigen der WAIT-Taste wird über eine Entprellschaltung ein Flipflop gesetzt, mit dessen Ausgang die Steuerbusleitung RDYIN (Anforderung von WAIT-Zyklen) aktiviert wird. Zwei Monoflops legen im Ruhezustand L-Pegel an die RDYIN-Leitung, worauf der Mikroprozessor WAIT-Zyklen einlegt. Durch einen Druck auf die ST-Taste (Step) wird von der Monoflopkette ein zeitverzögertes, kurzes READY-Signal an den Mikroprozessor gegeben (RDYIN wird H). Der Mikroprozessor legt einen Operationszyklus ein und wartet erneut auf einen ST-Tastendruck. Der WAIT-Zustand wird vom Mikroprozessor quittiert und bringt eine LED zum Leuchten.

Mit der R-Taste (Rücksetzen) kann das System auf den Programmanfang rückgesetzt werden.

Jeder Tastendruck wird über eine Verknüpfung mit einem NAND-Gatter 74LS30 an ein Monoflop gemeldet. Dieses Monoflop gibt einen Tongenerator frei, der einen kurzen Ton an einen Kleinlautsprecher abgibt. Der Tongenerator besteht aus zwei NAND-Gattern und einer Verstärkerstufe. Lediglich die Tasten DMA und WAIT werden akustisch nicht signalisiert, da bei ihnen bereits eine optische Kontrolle vorhanden ist.

Ein RC-Glied am Reset-Eingang des WAIT-Flipflops 74LS73 setzt dieses Flipflop beim Einschalten des Systems in den WAIT-Zustand. Der Mikroprozessor steht damit sofort am Anfang des Programmspeichers und befindet sich im WAIT-Zustand.

Reinhold Ruf

# Serielle Schnittstelle für 8080/8085

Zur Datenübertragung zwischen Mikrocomputer und peripheren Geräten wird häufig ein seriell arbeitendes Interface benötigt. Die hier beschriebene Schaltung ist ein solches Interface. Es dient der Umsetzung der vom Computer parallel angebotenen Daten in einen seriellen Datenstrom entsprechend den Anforderungen des peripheren Gerätes. Umgekehrt setzt es die vom E/A-Gerät kommenden seriellen Zeichen in parallele Zeichen um.

## Der USART

Die Schaltung verwendet den USART 8251A, der zu den meisten 8-bit-Mikroprozessoren kompatibel ist. Für den Betrieb einer seriellen Schnittstelle werden dem 8251A vom Mikroprozessor mit zwei Steuerworten (Kommandowort und Betriebsartwort) die Übertragungsbedingungen übergeben. Dazu adressiert der Mikroprozessor den 8251A und sendet die beiden Steuerworte in der angegebenen Reihenfolge.

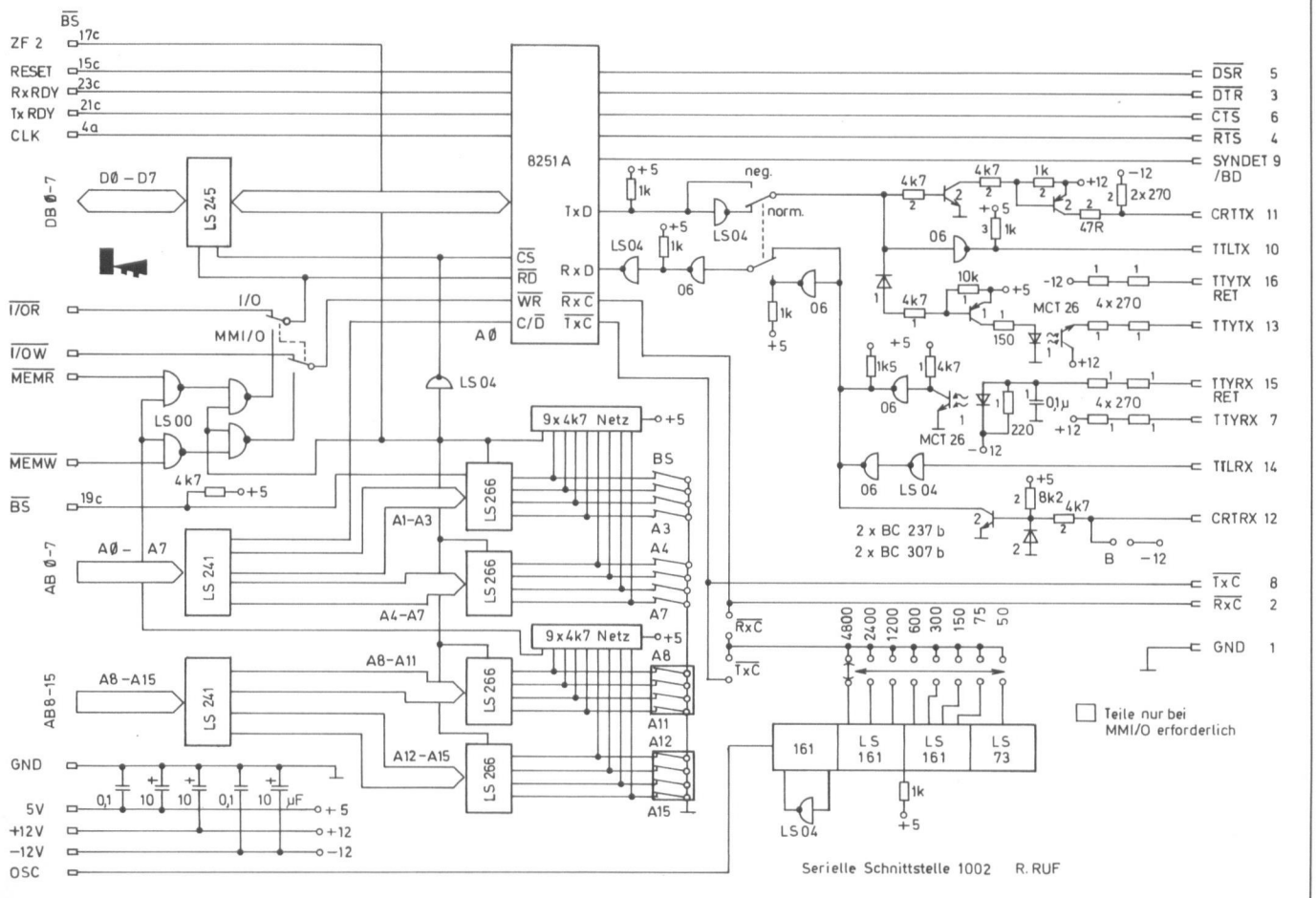
Um im adressierbaren E/A-Bereich eines 8080/85-Systems nur zwei Adressen durch die serielle Schnittstelle zu belegen, wurde beim vorliegenden Schaltungsentwurf ein einstellbarer Adressendekoder verwendet. Je nach Einstellung der DIL-Schalter A1 bis A7 wird bei Übereinstimmung mit der gesendeten Adresse die Karte freigegeben. Im freigegebenen Adressenbereich liegen nur die beiden Byte, die sich durch die Adresse A0 unterscheiden. Diese Adressenleitung wird zur Umsteuerung des 8251A von Steuerzeichen auf Daten und umgekehrt verwendet.

Das Ausgangssignal des Adressendekoders wird invertiert und dient so als $\overline{CS}$-Signal für den 8251A und den Datenbuspuffer 74LS245. Das nichtinvertierte Dekodersignal wird u.a. als Ausgangssignal $\overline{BS}$ verwendet.

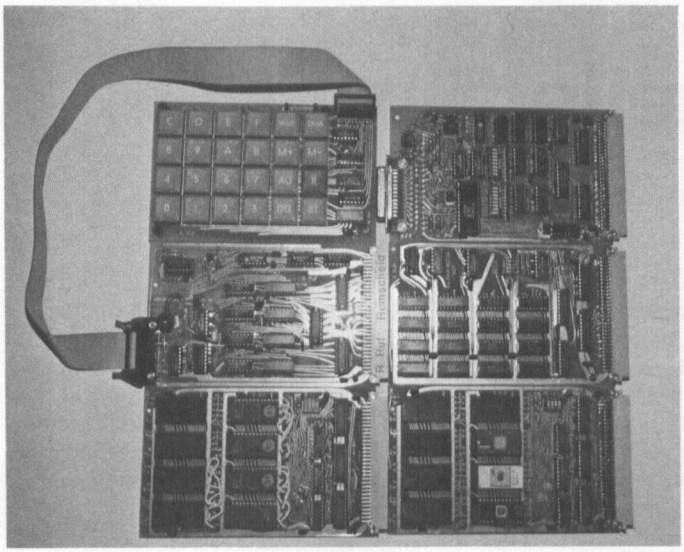
## Der BS-Ausgang

Mit dem Eingang $\overline{BS}$ kann ein externes Board-Select-Signal an den Dekoder angelegt werden. Mit einem L-Pegel an diesem Eingang und geschlossenem Schalter BS wird die Karte freigegeben. Wird der $\overline{BS}$-Eingang nicht benutzt, ist der Schalter BS zu öffnen und der Eingang frei zu lassen.

Die Richtung des Datenverkehrs wird




**Schaltung für die serielle Schnittstelle**



**Aufbau eines kompletten Computer-Kartensystems auf der Basis des 8085**

am 8251A und am Datenbuspuffer durch die Steuerbusleitungen I/O R bzw. I/O W eingestellt. Bei E/A-Aktivitäten des Mikroprozessors ist immer nur eine der beiden Steuerleitungen im L-Zustand. I/O R = L stellte den Datenbuspuffer in Leserichtung, I/O R = H stellt auf Schreibrichtung um. In den 8251A gelangen die vom Mikroprozessor gesendeten Zeichen jedoch erst bei gültiger Adresse und aktivem I/O W.

Um unabhängig von speziellen E/A-Befehlen zu sein und auch um Mikroprozessoren ohne spezielle E/A-Adressen verwenden zu können, kann die serielle Schnittstelle auch mit einer 16-bit-Adresse angesprochen werden. Dazu sind zusätzlich die Adressenleitungen A8 bis A15 erforderlich. Der Adressendekoder wird um die entsprechenden Leitungen erweitert, und die Karte wird über zwei benachbarte Adressen angesprochen.

Die gültige Adresse wird über die DIL-Schalter A1 bis A15 eingestellt, die Karte belegt dann die Adressen XXXX XXXX XXXX XXX0 für Daten und XXXX XXXX XXXX XXX1 für Steuerzeichen, wobei X die jeweilige Schalterstellung des Adreßwahlschalters bedeutet. Schalter offen = zugehörige Adresse 1, Schalter geschlossen = zugehörige Adresse 0.

Da in einem EPROM oft am Ende noch einige Bytes unbenutzt sind, liegt es nahe, diese Adressen anderweitig zu benutzen. Der voll einstellbare Adressendekoder des vorliegenden Schaltungsentwurfs macht diese unbenutzten Byte nutzbar. Dazu wird das Ausgangssignal $\overline{BS}$ herangezogen. Bei gültiger Schnittstellenadresse liegt am Ausgang $\overline{BS}$ H-Pegel. Dieser H-Pegel kann dazu benutzt werden, über den $\overline{BS}$-Eingang der ROM-Karte diese auszuschalten. Sie sind natürlich dann auf der ROM-Karte nicht mehr ansprechbar. Die Schreib-Lese-Steuerung wird in dieser Betriebsart über eine Verknüpfung der $\overline{MEMW}$ und $\overline{MEMR}$-Signale mit dem karteninternen SC erreicht.

Für flexiblen Einsatz ist auf der Karte ein Umschalter vorgesehen, mit dem zwischen normalem E/A-Betrieb (8-bit-Adresse) und Speicher-E/A-Betrieb (16-bit-Adresse) umgeschaltet werden kann. Bei normalem E/A-Betrieb müssen lediglich noch die beiden Vergleicher 74LS266 für die Adreßleitungen A8 bis A15 entfernt werden.

Die Takterzeugung des 8251A kann wahlweise von einem Taktgenerator (Baud-Rate-Generator) auf der Karte oder von einem externen Taktgenerator erfolgen. Dazu sind die Eingänge $\overline{RxC}$ und $\overline{TxC}$ des 8251A auf den Interfacestecker geführt. Der interne Baud-Rate-Generator wird vom OSC-Signal des Taktgeberbausteins im 8080-System gespeist. Er teilt diese Frequenz je nach Einstellung des Schiebeschalters auf die dem Übertragungstakt entsprechende Frequenz herunter. Die erste Teilerstufe setzt das Eingangssignal auf 15 : 1, der Ausgang 4800 zum Taktschiebeschalter ist von dieser Frequenz nochmals um 4 :1 untersetzt. Die angegebenen Übertragungsgeschwindigkeiten gelten bei einem programmierten 8251A-internen Teilerverhältnis von 64 : 1 und einer Quarzfrequenz des OSC-Eingangs von 18,432 MHz. Die nachfolgenden Teilerstufen untersetzen jeweils 2 : 1. Bei Verwendung des internen Taktgenerators ist sein Ausgang mit den beiden Takteingängen $\overline{RxC}$ und $\overline{TxC}$ des 8251 zu verbinden.

Der Anschluß des E/A-Gerätes erfolgt je nach Typ des Gerätes über unterschiedliche Schnittstellen. Hier stehen drei verschiedene Arten von seriell arbeitenden Schnittstellenleitungen zur Verfügung: 20-mA-Stromschleife für z.B. TTY, V24 Spannungsschnittstelle für z.B. Datensichtgerät und eine TTL-Schnittstelle. Das Ausgangssignal des 8251A kann über einen Umschalter im Bedarfsfall negiert werden, gleichzeitig wird dann auch der Empfangspegel für alle Schnittstellenleitungen negiert.

Sendeseitig ist die 20-mA-Stromschleife über einen Schalttransistor und einen Optokoppler realisiert. Der Linienstrom wird von der seriellen Schnittstelle eingespeist, so daß auf der E/A-Geräteseite keine Spannung benötigt wird. Als Empfänger ist dort z.B. ein Empfangsmagnet vorhanden.

Die Empfangsseite der Stromschleife hat ebenfalls einen Optokoppler im Linienstromkreis. Sättigungserscheinungen in der LED werden durch einen 220-Ω-Widerstand verhindert.

Bei nicht benutzter Stromschleife ist der Ruhezustand der Linie herzustellen, d.h., zwischen TTYRX und TTYRXRET ist eine Brücke einzubauen. Dies kann z.T. im Gerätestecker erfolgen. Soll der Linienstrom erhöht werden, weil z.B. die 20 mA nicht ausreichen, um den Empfangsmagneten im Fernschreiber zu betätigen, so kann dies durch Überbrücken eines der beiden 270-Ω-Widerstände in den Linienzuleitungen geschehen.

Die V24-Schnittstelle gibt im Ruhezustand −12 V auf die Leitung. Startpotential wird durch Schalten der +12 V auf den Spannungsteiler aus 47 Ω und 2 × 270 Ω erzeugt.

Auf der Empfangsseite setzt ein einfacher Transistorschalter mit Schutzdiode die negativen bzw. positiven Empfangsspannungen in TTL-Pegel um. Wenn dieser Teil der Schaltung aufgebaut ist und nicht benutzt wird, so ist die Brücke B nach −12 V einzubauen, damit der Transistor gesperrt wird. Dies entspricht dem Ruhezustand der V24-Leitung.

Die TTL-Schnittstelle ist sendeseitig mit offenem Kollektor ausgelegt. Sollte in der Gegenstelle bereits ein Pull-up-Widerstand enthalten sein, kann er hier entfallen. Die Empfangsseite ist als LS-Eingang realisiert und bietet keine Besonderheiten.

Alle Eingänge und Ausgänge der Schnittstellenleitungen sind zum E/A-Gerätestecker hin nicht vorverdrahtet. Je nach vorliegenden Bedürfnissen kann jeder Anschluß der 25poligen IEC-Buchse mit der entsprechenden Schnittstellenleitung verbunden werden. Dies gilt auch für die übrigen Leitungen wie Masse, Takt, Synchronisation und Modemsteuerung.

Norbert Linder

# Schnelle multiplizierende und dividierende arithmetisch-logische Einheit

Ein bei allen Mikrocomputersystemen häufig auftretendes Problem ist die lange Rechenzeit für rein softwaremäßig realisierte arithmetische Operationen. Dies wirkt sich besonders bei Multiplikationen und Divisionen aus, die auch Voraussetzung für die Berechnung aller komplexeren Funktionen wie SIN(X) oder LG(X) sind. So liegt z.B. die Ausführungszeit für eine Gleitpunktmultiplikation im 32-bit-Format meistens schon im Millisekundenbereich. Bei umfangreicheren mathematischen Problemstellungen treten daher ziemlich lange Programmlaufzeiten auf, unter Umständen können manche Aufgaben wegen des hohen Zeitaufwandes gar nicht oder nur unbefriedigend gelöst werden. Ein Beispiel dafür ist die numerische Integration, die zur Erzielung einer ausreichenden Genauigkeit sehr viele Rechenschritte erfordert.

Eine Lösung des Problems ist die Verwendung von Arithmetikprozessoren. Von Vorteil ist, daß diese Bausteine neben den mathematischen Grundoperationen auch die wichtigsten komplexeren Funktionen ausführen können; als Nachteil steht der hohe Preis gegenüber.

Eine Alternative dazu kann die hier beschriebene Schaltung bilden. Sie läßt sich mit geringerem finanziellen Aufwand aufbauen, bietet aber nicht den Komfort eines Arithmetikprozessors.

Rein hardwaregesteuert kann sie sehr schnell einfache Multiplikationen und Divisionen ausführen; zusätzlich stellt sie alle Funktionen, die die verwendete ALU (74181) durchführen kann, zur Verfügung. Darunter fallen u.a. Addition und Subtraktion mit und ohne Übertrag, Inkrementieren, Dekrementieren und zahlreiche logische Verknüpfungen. Die Wortlänge der parallel verarbeiteten Daten kann praktisch frei gewählt werden, so daß die Einheit ohne weiteres für sämtliche Mantissenoperationen bei Gleitpunktzahlen beliebigen Formats verwendbar ist. Zusammen mit entsprechender Software können damit in einem Mikrocomputersystem alle Funktionen eines Arithmetikprozessors mit etwa gleicher Rechenzeit realisiert werden.

## Die Schaltungsprinzipien

Da die Gesamtschaltung relativ umfangreich ist, soll hier zunächst anhand vereinfachter Darstellungen die Funktionsweise der Schaltung bei den verschiedenen Operationsarten erklärt werden.

Man kann drei sinnvolle „Zustände" unterscheiden, die abhängig von den Werten der M- und D-Bit im Befehlswort sind. Je einer dieser Zustände ist der Multiplikation, der Division und den restlichen „einfachen" ALU-Operationen zugeordnet.

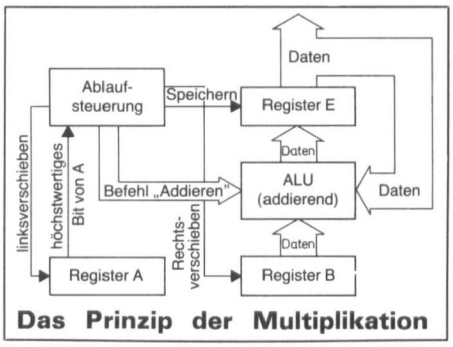
## Eine Multiplikation

Dem Rechenvorgang liegt das „normale" Multiplikationsverfahren für Dualzahlen zugrunde: Der Reihe nach werden alle Stellen des Operanden A abgefragt und abhängig von deren Wert die jeweils entsprechend verschobenen Darstellungen des Operanden B aufaddiert.

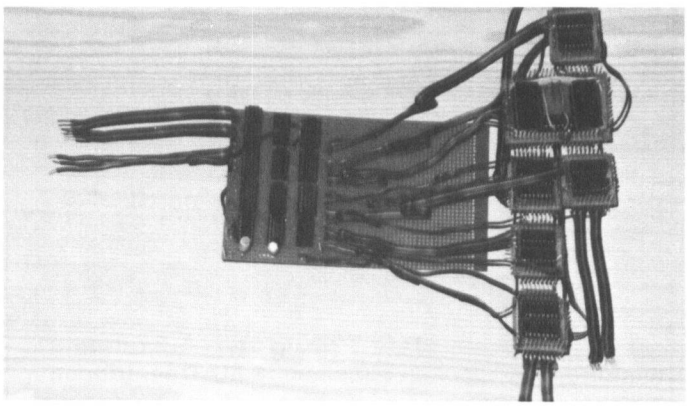
Dazu müssen der als Addierer arbeitenden ALU die parallelen Daten des Registers B und des Registers E zugeführt werden. Die Daten des Registers A gelangen nicht an die ALU; es wird von der Ablaufsteuerung nur das jeweils höchstwertige Bit 0 abgefragt.

Zu Beginn der Operation wird das Register E gelöscht. Der Addierer liefert an seinem Ausgang also den Wert des Operanden B. Wenn das Bit 0 von A auf 1-Signal 1 ist, veranlaßt die Ablaufsteuerung das Abspeichern des Additionsergebnisses im Register E; andernfalls findet dieser Vorgang nicht statt. Danach wird der Operand A um eine Stelle nach links, der Operand B um eine Stelle nach rechts verschoben. Dadurch rückt die zweite Stelle von A als höchstwertiges Bit nach. Am ALU-Ausgang steht jetzt die Summe aus dem vorher in E abgespeicherten Zwischenergebnis und dem verschobenen Operanden B. Das Additionsergebnis wird wieder abhängig vom neuen Bit 0 von A entweder abgespeichert oder nicht.

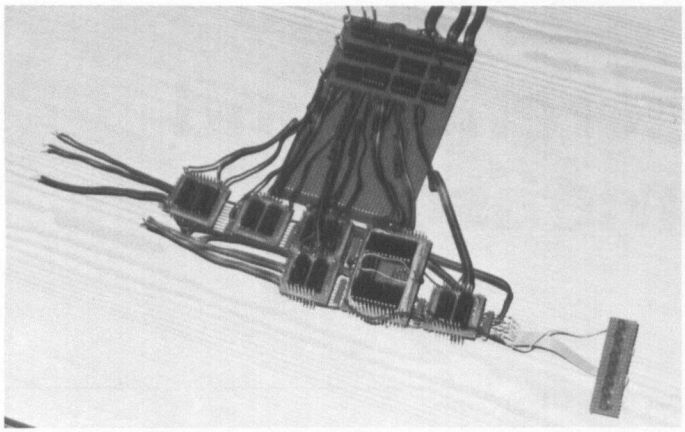
Dieser Vorgang wiederholt sich so oft, wie die Anzahl der Stellen der Operanden beträgt. Überlegt man sich die hier dargestellte Arbeitsweise an einem praktischen Beispiel, so kann man erkennen, daß sie sich von den üblicherweise auf dem Papier durchgeführten Berechnungen nur insofern unterscheidet, daß bei jedem Rechenschritt eine Zwischensumme erzeugt wird, während man bei „manuellen" Multiplikationen die Zwischenprodukte am Schluß addiert.



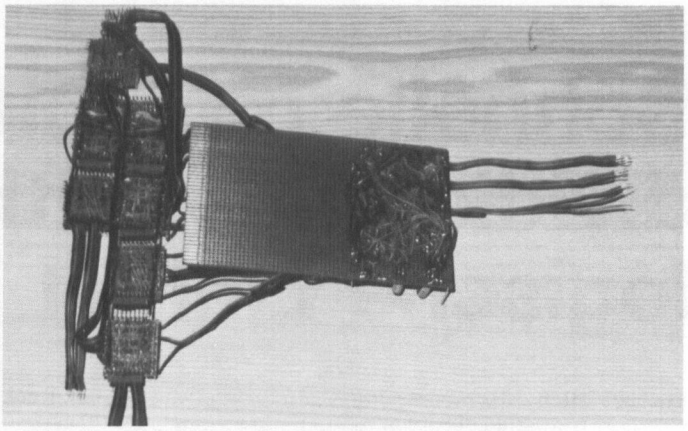

**Das Prinzip der Multiplikation**



**Schaltung der Arithmetikeinheit von oben. Auf der linken Seite die Ablaufsteuerung, rechts die als „Minimoduln" aufgebauten Register-, Multiplexer- und ALU-Einheiten.**



**Schaltung der Arithmetikeinheit im Testaufbau; Leuchtdioden am Ausgang**



**Testschaltung der Arithmetikeinheit von unten**

## Die Division

Auch die Division erfolgt nach dem allgemein gebräuchlichen Prinzip: Operand B wird vom Operanden A abgezogen. Ist die Differenz positiv, wird sie als neuer Operand A übernommen, gleichzeitig wird an die Ergebnisdarstellung eine 1 angehängt. Falls die Differenz negativ ist, behält A seinen alten Wert bei, am Ergebnis wird eine 0 angetragen. Danach wird der gleiche Vorgang mit dem um eine Stelle nach rechts verschobenen Operanden B wieder ausgeführt.

Die Datenleitungen von A und B sind jetzt mit der subtrahierenden ALU verbunden (A – B). Die ALU-Ausgangsdaten werden an die Paralleleingänge des Registers A geführt.

Das Vorzeichen des Subtraktionsergebnisses kann aus dem Zustand des ALU-Carry-Bit abgeleitet werden. Daher wird es (invertiert) an den seriellen Eingang des Registers E und an die Ablaufsteuerung geleitet.

Nach dem Start der Operation erfolgt im E-Register ein Schiebevorgang nach links, wodurch das (invertierte) Carry-Bit in das Register übernommen wird. Außerdem speichert das Register A das Subtraktionsergebnis der ALU, falls der Wert des Carry-Bit anzeigt, daß es positiv ist. Anschließend wird der Inhalt des Registers B um eine Stelle nach rechts verschoben, und der Vorgang beginnt erneut. Er wiederholt sich wie bei der Multiplikation so oft, wie die Operanden Stellen haben.

## Eine „einfache" ALU-Operation

Bei der Ausführung der ALU-eigenen arithmetischen und logischen Funktionen sind die Datenleitungen beider Operanden an die entsprechenden ALU-Eingänge geführt, ebenso ist das Register E an die Ausgänge der ALU angeschlossen. Über das Befehlswort bestimmt die Ablaufsteuerung die Auswahl der ALU-Funktion. Bei der Operation wird lediglich das Register E am Anfang gelöscht und dann mit dem ALU-Ausgangssignal wieder beschrieben.

## Die Erweiterung

Die Schaltung ist für die Verarbeitung von Gleitpunktmantissen entwickelt worden. Dazu muß die Länge aller Register nur der Mantissenlänge entsprechen.

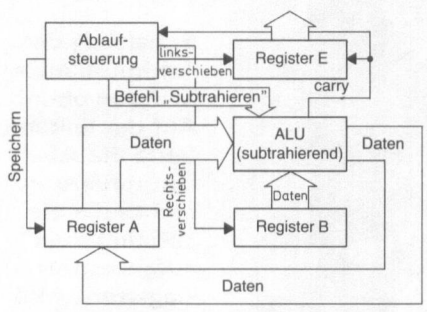
Es ist jedoch der höheren Rechengenauigkeit wegen sinnvoll, mit einigen weiteren Stellen zu rechnen. Deshalb sind alle Einheiten, die die schaltungsintern verlaufenden Paralleldaten verarbeiten, aufgeteilt in einen „Wortteil", der die Länge der Mantissen umfaßt, und in einen „Erweiterungsteil", der die zusätzlichen Stellen zur Verfügung stellt. Vom Mikrocomputer aus können bei den Registern A und B nur die Wortteile angesprochen werden. Für das Register E gilt ähnliches; es kann jedoch durch entsprechende Schaltungsauslegung auch eine Auslesung des gesamten Registers vorgesehen werden, wodurch die höhere Genauigkeit auch vom Mikrocomputer genutzt werden kann. Die Länge des Wortteils entspricht der Länge der Gleitpunktmantisse, die des Erweiterungsteils kann der Anwender selbst wählen.

## Die Gesamtschaltung

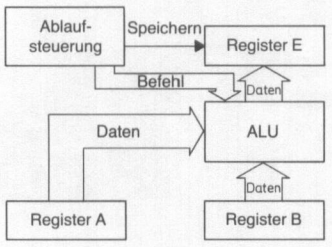
Die Gesamtschaltung zeigt den Aufbau der Elektronik. Der Übersicht halber sind die Register-, Multiplexer- und ALU-Einheiten sowie die Ablaufsteuerung als Blöcke dargestellt, deren detaillierte Innenschaltungen in separaten Bildern gezeigt werden. Außerdem wurden die parallelen Datenleitungen zwischen den Einheiten durch Pfeile ersetzt.

Zum Verständnis der Schaltung sollen zunächst die einzelnen Einheiten, mit Ausnahme der Ablaufsteuerung, besprochen werden. Sie verarbeiten alle die schaltungsintern parallel verlaufenden Daten und bestehen daher aus einem Wort- und einem Erweiterungsteil. In den Bildern ist die Trennung durch eine gestrichelte Linie kenntlich gemacht.

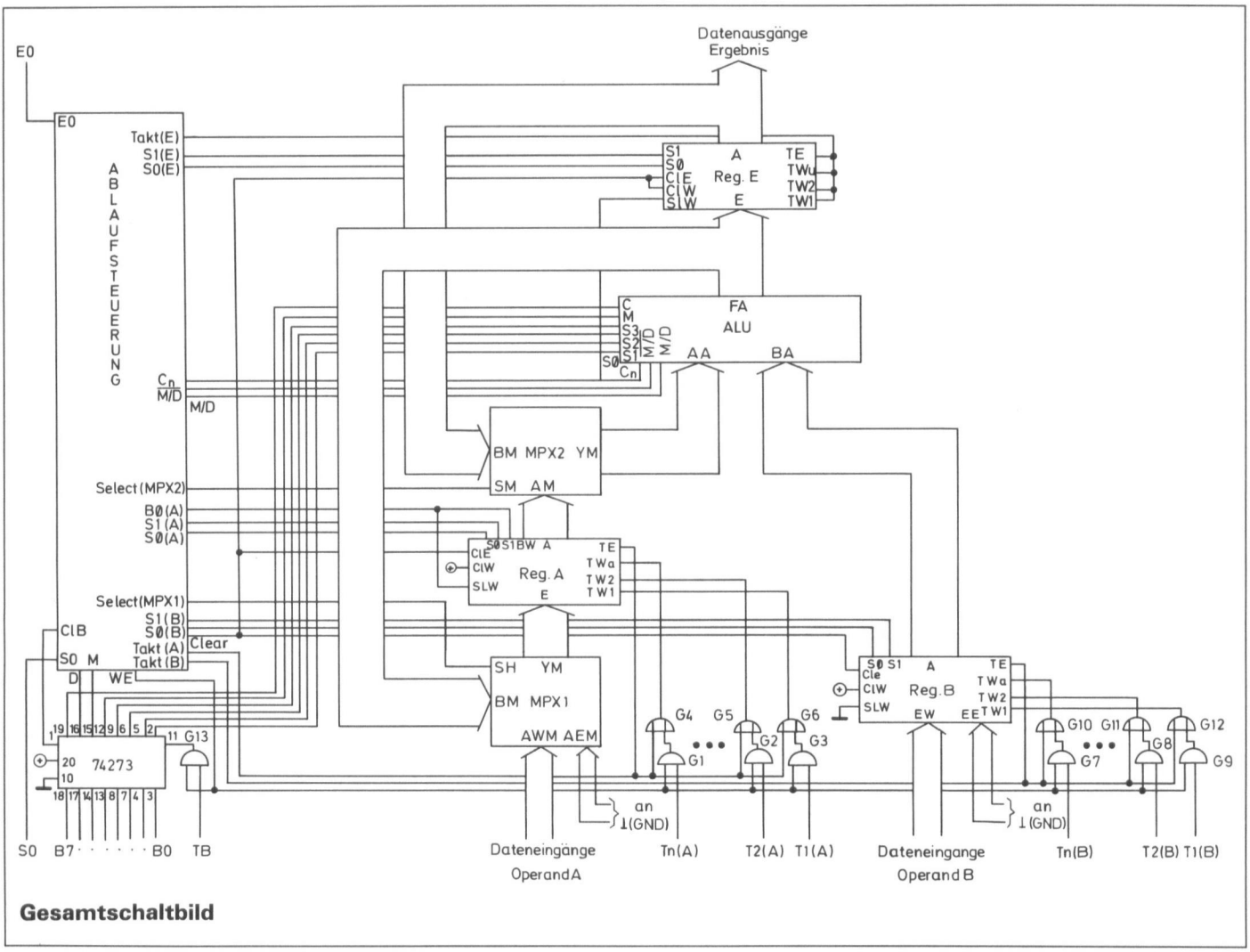
Da die Einheiten auf beliebige Länge ausgebaut werden können, wurde versucht, eine allgemeine Darstellung für sie zu finden. Dazu sind jeweils zwei Schaltkreise im Wort- und Erweiterungsteil in ihrer Zusammenschaltung gezeichnet. In die Lücke zwischen diesen und dem dritten können weitere Schaltkreise auf die gleiche Weise wie die beiden ersten angeschlossen werden. Der umrandete Teil kann in der Gesamtschaltung als „Black-Box" betrachtet werden; die Bezeichnungen der herausgeführten Ein- und Ausgänge entsprechen denen in den Detailschaltplänen.




**Das Prinzip der Division**




**Das Prinzip der Ausführung „einfacher" ALU-Operationen**




**Gesamtschaltbild**

## Die Registereinheit

Alle drei benötigten Register A, B und E sind aus der gleichen Registereinheit aufgebaut.

Sie besteht aus 4-bit-Universalschieberegistern 74194, die, getrennt nach Wort- und Erweiterungsteil, so zusammengeschaltet sind, daß, neben dem Speichern natürlich, Rechtsverschieben des gesamten Registers und Linksverschieben der einzelnen Teile möglich ist. Dazu ist der „Shift Left Seriell Input" eines jeden Schieberegisters jeweils mit dem $Q_D$-Ausgang des vorherigen, der „Shift Right Seriell Input" mit dem $Q_A$-Ausgang des nachfolgenden Registers verbunden; an der Verbindungsstelle Wort-Erweiterung wird der Eingang zum Linksschieben als SLW(R) nach außen geführt.

Weiterhin sind extern zugänglich die Eingänge S0 und S1 (Auswahl des Arbeitsmodus: Speichern, Rechts-, Linksschieben), die Clear- und die Takteingänge.

Die Clear-Eingänge der einzelnen IC sind getrennt nach Wort-(ClW) und Erweiterungsteil (ClE) zusammengefaßt; bei den Takteingängen (TE, TW1, TW2 ···TWn) kann zusätzlich jedes Schieberegister im Wortteil einzeln angesprochen werden. Am Ausgang B0(R) kann das höchstwertige Bit der Registereinheit abgegriffen werden.

## Die Multiplexereinheit

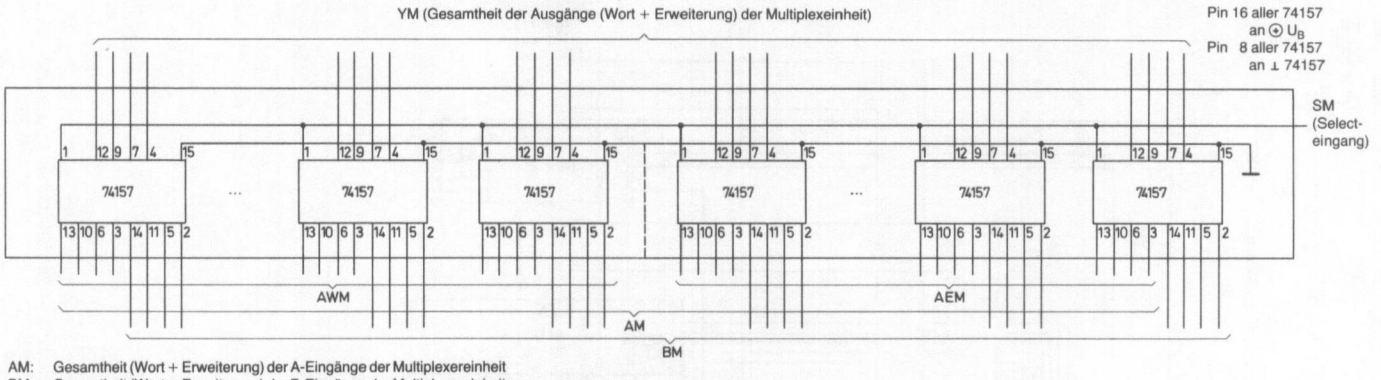
Für die Schaltung werden zwei identische Multiplexereinheiten benötigt. Es werden die TTL-Schaltkreise 74157 verwendet. Der Aufbau ist sehr einfach und direkt anhand des Schaltbildes verständlich: Die Daten von einem der Eingänge AM und BM werden je nach dem Pegel an SM (0 für AM, 1 für BM) an den Ausgang durchgeschaltet. Zwischen Wort- und Erweiterungsteil besteht kein Unterschied.

## Die ALU-Einheit

Die ALU-Einheit besteht aus den 4-bit-ALU-Bausteinen 74181, die parallel geschaltet werden.

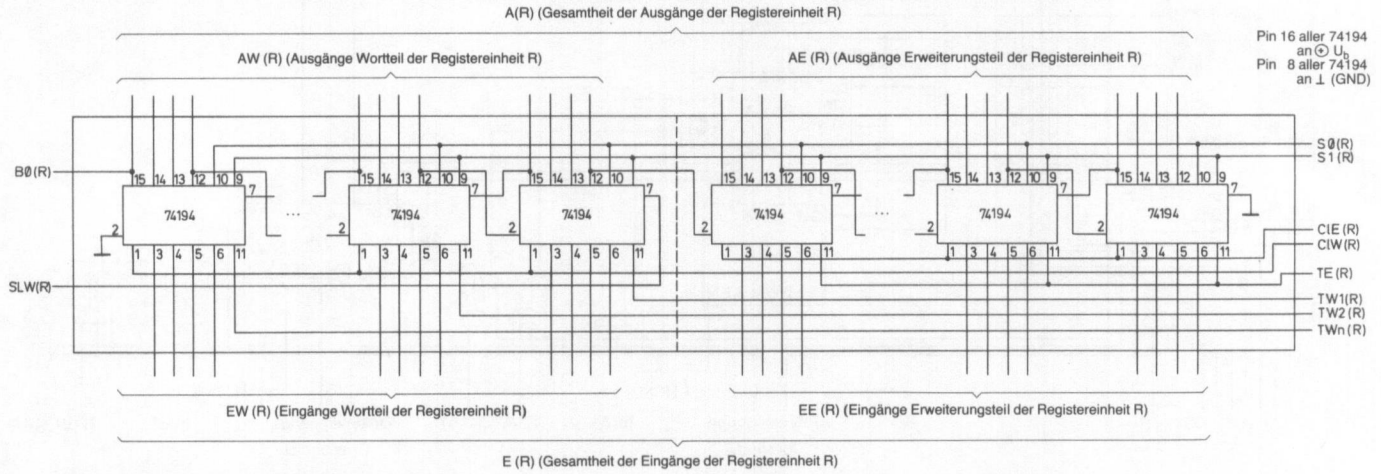
Für die Übertragungsbildung gibt es zwei Möglichkeiten: Einmal die serielle Übertragungsbildung, bei der der Carry-Ausgang einer 74181 einfach mit dem Carry-Eingang der nächsthöheren verbunden wird. Diese Methode reicht für kleine Einheiten aus; ab einer Länge von 16 bit wird sie jedoch zu langsam. Dann empfiehlt sich die parallele Übertragungsbildung. Dazu wird zusätzlich die Übertragungseinheit 74182 benötigt, und zwar für Blöcke von vier ALU-Bausteine 74181 je eine, für vier dieser Blöcke wieder je eine usw. Zu beachten ist, daß die Übertragungsbildung für Wort- und Erweiterungsteil getrennt erfolgen muß.

An deren Verbindungsstelle übernimmt die gezeigte Logik aus den zwei UND- und ODER-Gatter die Übertragserzeugung für die erste ALU des Wortteils. Falls das $\overline{M/D}$-Signal auf 1 ist und damit anzeigt, daß weder eine Multiplikation noch eine Division erfolgen soll, wird das im Befehlswort enthaltene Bit für den Übertrag über den $C_0$-Eingang direkt an die erste Wort-ALU geschaltet. Im anderen Fall gelangt es nur an die erste ALU des Erweiterungsteils; zwischen Wort und Erweiterung wird der Carry seriell übertragen (M/D auf 1-Signal). Grund dafür ist, daß nur bei Multiplikation und Division der Erweiterungsteil mit benutzt wird.




AM: Gesamtheit (Wort + Erweiterung) der A-Eingänge der Multiplexereinheit
BM: Gesamtheit (Wort + Erweiterung) der B-Eingänge der Multiplexereinheit
AWM: A-Eingänge Wortteil der Multiplexereinheit
AEM: A-Eingänge Erweiterungsteil der Multiplexereinheit

**Detailschaltung der Multiplexereinheit**




**Detailschaltung der Registereinheit**

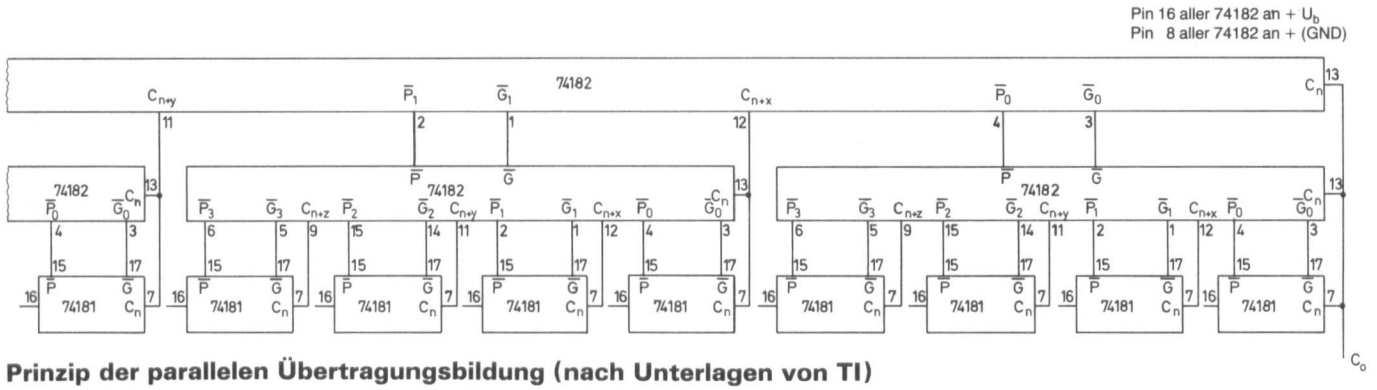
Die Eingänge S0, S1, S2, S3 und M sind für die Funktionsauswahl verantwortlich.
$C_n$ ist der Übertragungsausgang der gesamten ALU-Einheit. Durch den Inverter wird er für die Verwendung bei der Division angepaßt ($C_n = 0$, falls Subtraktionsergebnis negativ, $C_n = 1$, wenn positiv).

## Der Aufbau der Gesamtschaltung

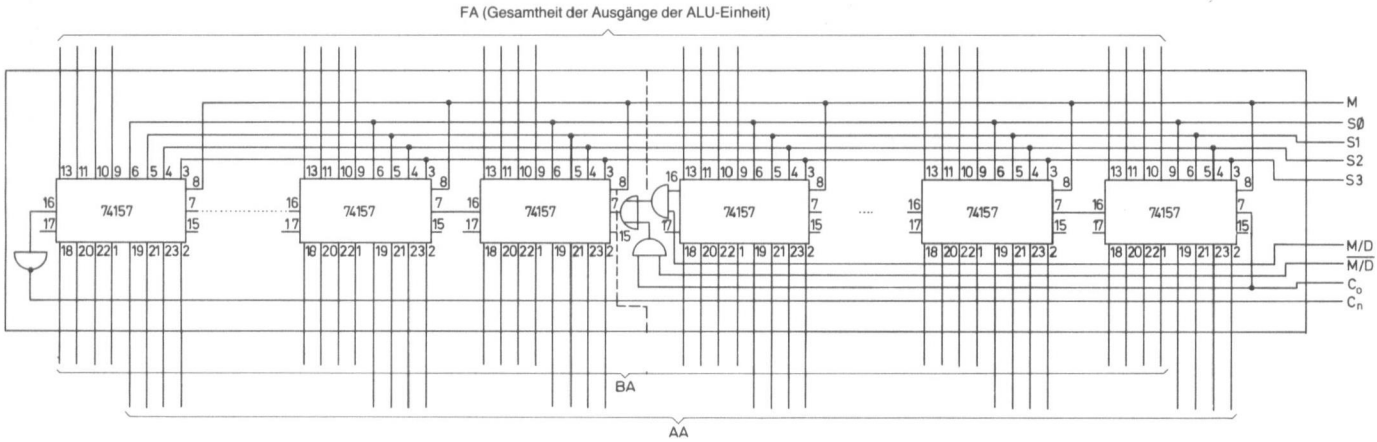
Die Gesamtschaltung besteht im wesentlichen aus den schon besprochenen Einheiten, der Ablaufsteuerung, dem Befehlsregister und der Logik zur Taktsteuerung der Register A und B, bestehend aus den Gattern G1 ··· G13.
Die Multiplexer MPX1 und MPX2 stellen die verschiedenen für die Multiplikation und die Division erforderlichen Anschaltungsweisen des Registers A an die übrigen Einheiten her.
In dem mit dem IC 74273 aufgebauten Register steht der 8-bit-Befehlskode für die gewünschte Operation. Befehl und Operanden können durch Taktimpulse an TB, T1(A) ··· Tn(A) und T1(B) ··· Tn(B) abgespeichert werden, solange keine Operation ausgeführt wird. Die Eingangsdaten für die Register A und B werden dabei an die Worteingänge gelegt (bei A an die des Multiplexers MPX1); die Eingänge der Erweiterungsteile sind mit Masse verbunden.
Eine positive Flanke am Eingang S0 bewirkt den Start des Operationsablaufs, das E0-Signal kennzeichnet das Ende.

## Der zeitliche Ablauf der Operationen

Nach jeder vollständigen Ausführung einer Rechenoperation werden alle Flipflops FF1 ··· FF4 in der Ablaufsteuerung und das Befehlsregister zurückgesetzt.
Damit sind die Eingänge S0 und S1 der Register A und B beide auf 1-Signal, die Einheiten also aufs Speichern vorbereitet. Außerdem führt die WE-(„Write Enable"-)Leitung einen H-Pegel. Der Multiplexer MPX1 schaltet wegen des L-Pegels am Select-Eingang die Daten vom Eingang A an den Ausgang durch. Vom Mikrocomputer können nun die Operanden und der Befehl in die Register geschrieben werden. Dazu müssen, während die Daten an den Eingängen liegen, Taktimpulse mit positiver Flanke an die Takteingänge T1(A)···Tn(A), T1(B)··· Tn(B) und TB gesendet werden. Dabei ist zu beachten, daß bei Multiplikation und Division das Befehlswort zuletzt abgespeichert wird, da mit dessen Übernahme der Multiplexer MPX1 keine Daten vom A-Eingang mehr weitergibt (bei der Division) und das Register B auf Schiebebetrieb umschaltet.
Eine positive Flanke an SØ startet nun den Rechenvorgang. Dabei wird FF1 gesetzt, das durch den Pegelwechsel an $\overline{Q}$ auch FF2 taktet. Während der beim Setzen von FF2 auftretenden Verzögerungszeit wird der Ausgang des ODER-Gatters N8 auf Ø-Signal und löscht damit die gesamte Registereinheit E und die Erweiterungsteile von A und B. Sobald $\overline{Q}$ von FF2 auf Ø-Signal geworden ist, wird FF1 wieder in seinen Ausgangszustand gebracht. Außerdem werden nun über die WE-Leitung (Ø-Signal) externe Taktimpulse gesperrt.
Der positive Pegel am Q-Ausgang von FF2 gibt das Flipflop FF3 frei. Mit jedem vom Taktgenerator, bestehend aus den




**Prinzip der parallelen Übertragungsbildung (nach Unterlagen von TI)**




**Detailschaltung der ALU-Einheit**

Invertern N1, N2 und N3, kommenden Impuls werden jetzt dessen Ausgänge umgeschaltet. Dadurch entsteht ein symmetrischer Zweiphasentakt mit Q = 1 als erster und $\overline{Q}$ = 1 als zweiter Taktphase. FF4 liefert erst nach dem ersten Taktsignal einen H-Pegel an N18 und N22 und verhindert dadurch, daß bereits vor Operationsstart, während des Einschreibens des Befehls, über N15 oder N21 Taktimpulse an die Registereinheiten A und B gelangen können.

Der Q-Ausgang von FF3 wird an den Eingang des IC 74393 geführt. Die zwei in diesem IC enthaltenen 4-bit-Binärzähler sind zu einem 8-bit-Zähler zusammengeschaltet, der seinen Zählerstand mit jeder negativen Flanke am Eingang erhöht.

Eine „einfache" ALU-Operation wird innerhalb eines Taktzyklus ausgeführt. Da die Leitungen M und D beide L-Pegel führen, ist das durch N19 und N20 erzeugte $\overline{M/D}$-Signal auf 1-Signal. Bereits nach der ersten negativen Flanke am Zählereingang schalten daher N5 und N6 durch. Dadurch werden der Zähler und gleichzeitig über N7 das Befehlsregister, die Flipflops FF2 und FF4 und als Folge davon auch FF3 zurückgesetzt. Der Pegel des EØ-Ausgangs wechselt von L auf H und zeigt das Operationsende an. Dieser Zustand wird so lange aufrechterhalten, bis eine neue positive Flanke an FF1 eintrifft.

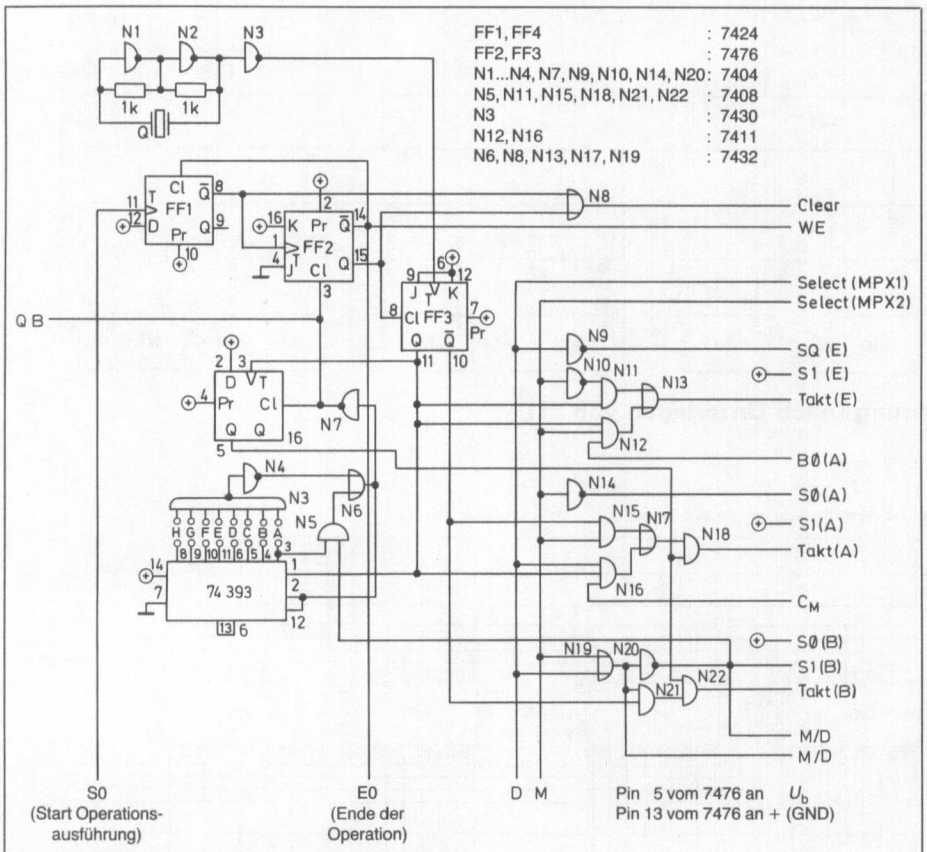
Bei einer Multiplikation oder Division soll die Reset-Routine erst beginnen, wenn der Zählerstand den Wert der Länge der Operanden erreicht hat. Dazu werden diejenigen Zählerausgänge, die diesen Wert binär darstellen, an das NAND-Gatter N3 geführt, alle anderen Gattereingänge werden auf H-Pegel gelegt (z.B.: Mantissenlänge 24 bit = 00011000, d.h. die Ausgänge D und E des Zählers an N3, alle anderen Anschlüsse von N3 an + $U_b$). Da N5 gesperrt ist, wird der Reset erst eingeleitet, wenn der Zähler den so vorgegebenen Stand erreicht. Innerhalb der eben beschriebenen Initialisierungs- und Reset-Routine laufen die eigentlichen Rechenoperationen ab. Die folgenden Erläuterungen zeigen die schaltungstechnische Realisierung der am Anfang theoretisch dargestellten Arbeitsweise.

## Der Multiplikationsablauf

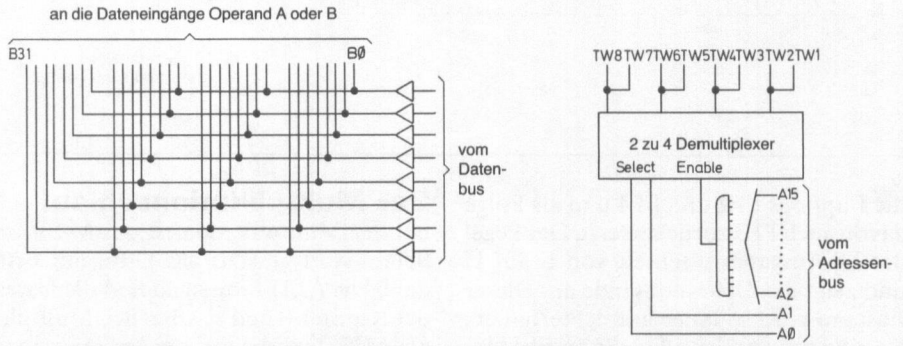
Bei der Multiplikation ist das M-Bit im Befehlswort gesetzt, das D-Bit auf Ø-Signal. Die ALU-Einheit addiert die Daten der Register B und E. Über den Multiplexer MPX2 wird diese am Anfang prinzipiell beschriebene Verschaltung der Datenwege hergestellt. Der Rechenvorgang läuft folgendermaßen ab: Über die S0- und S1-Eingänge ist das Register E aufs Speichern, A aufs Links- und B aufs Rechtsschieben vorbereitet.

Ist das höchstwertige Bit von A(B0(A)) auf 1-Signal, dann schalten während der ersten Taktphase die Gatter N12 und N13 durch. Da N11 gesperrt ist, erscheint eine positive Flanke am Takteingang der Registereinheit E. Diese übernimmt das Additionsergebnis vom Ausgang der ALU. Mit der zweiten Taktphase gelangen positive Impulse über die Gatter N15, N17, N18 und N19, N21 und N22 an alle Takteingänge von A und B und bewirken das Links- bzw. Rechtsverschieben der gesamten Registereinheiten.

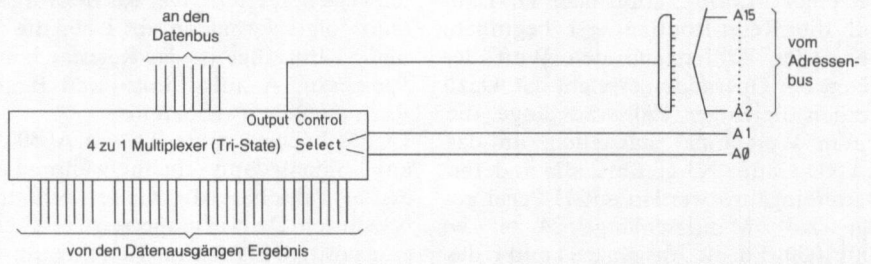
Dieser Vorgang wiederholt sich mit dem folgenden Taktzyklus so lange, bis der Reset eingeleitet wird.




**Schaltung der Ablaufsteuerung**




**Anschluß der Registereinheiten A und B an einen Mikrocomputer (Prinzip)**




**Anschluß der Registereinheit E an einen Mikrocomputer (Prinzip)**

Einen gewissen Vorteil bietet die Rückführung von B0(A) auf den SLW-Eingang: Der Operand A bleibt bei der Rechnung erhalten und kann sofort weiterverwendet werden.

## Der Divisionsablauf

Bei der Division ist das D-Bit auf 1-Signal, das M-Bit auf Ø-Signal. Dadurch werden die Ausgangsdaten der ALU über den Multiplexer MPX1 an die Eingänge der Registereinheit A gelegt. Das Register E ist auf Linksschieben umgeschaltet; es kann nur seriell den $C_n$ der subtrahierenden ALU-Einheit über den Eingang SLW übernehmen. Register A ist durch S0 und S1 aufs Speichern, Register B aufs Rechtsschieben vorbereitet.

Mit der ersten Taktphase wird die Einheit E über N10, N11 und N13 getaktet und übernimmt den Carry der ALU. Gleichzeitig schalten N16, N17 und N18 durch, falls der Carry auf 1-Signal, das Subtraktionsergebnis also positiv ist. Dadurch wird es im Register A abgespeichert. Dieser Vorgang entfällt bei negativer Differenz. Während der zweiten Taktphase wird nur noch das Register B nach rechts verschoben.

## Ein „einfacher" ALU-Operationsablauf

Weder das M-Bit noch das D-Bit im Befehlswort darf gesetzt sein, so daß die Registerdaten direkt an die Engänge der ALU gelangen. Außerdem sind alle Registereinheiten auf Speichern geschaltet. Während der einzigen Taktphase übernimmt das Register E, getaktet über N11 und N13, das an den Ausgängen der ALU stehende Verknüpfungsergebnis. Beide Operanden bleiben erhalten.

## Hinweise zum Nachbau der Schaltung

Die Schaltung ist vollständig in TTL-Technik ausgeführt. Man benötigt daher die übliche 5 V-Betriebsspannung. Dabei ziehen besonders größere Ausbaustufen relativ viel Strom. Das ist vor allem dann zu beachten, wenn das vorhandene Mikrocomputer-Netzteil zur Versorgung verwendet werden soll. Eventuell ist eine eigene Stromversorgungseinheit für die Schaltung vorzusehen.

Bei der praktischen Ausführung der Schaltung ist es empfehlenswert, die modulare Struktur beizubehalten und entweder Register-, Multiplexer- und ALU-Einheiten getrennt voneinander aufzubauen oder jeweils den die Daten verarbeitenden Teil für eine bestimmte Länge auf einer Platine unterzubringen. Letztere Methode hat den Vorteil, daß eine nachträgliche Erweiterung leicht möglich ist. Auf jeden Fall sollten die Ablaufsteuerung und das Befehlsregister auf eine eigene Platine gelötet werden.

Die maximale Rechengeschwindigkeit hängt von der verwendeten TTL-Technik ab. Meistens werden es LS-Schaltkreise sein; für die ALU und die Übertragseinheit ist es jedoch sinnvoll, reine Schottky-IC einzubauen. Auch die maximale Taktfrequenz wird damit hauptsächlich durch die TTL-Familie bestimmt, weniger durch die Größe der Einheiten. Bei LS-Schaltkreisen funktioniert die Schaltung noch einwandfrei bei 10 MHz, mit Schottky-ALU-Bausteinen sind auch 20 MHz möglich. Für besonders hohe Rechengeschwindigkeiten können die an-

D M

| $C_{\emptyset}$ | | | M | S3 | S2 | S1 | S0 |
|---|---|---|---|---|---|---|---|
| B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |

Aufbau des Befehlswortes

Multiplikation:

| H | L | H | L | H | L | L | H |
|---|---|---|---|---|---|---|---|
| B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |

Division:

| L | H | L | L | L | H | H | L |
|---|---|---|---|---|---|---|---|
| B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |

„Einfache" ALU-Operationen:

| $C_{\emptyset}$ | L | L | M | S3 | S2 | S1 | S0 |
|---|---|---|---|---|---|---|---|
| B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |

gegebenen Werte noch gesteigert werden. Dazu kann für Versuche der Quarz Q durch einen Kondensator ersetzt werden.

## Der Anschluß an den Mikrocomputer

Beim Anschließen der Schaltung an das Mikrocomputersystem gibt es prinzipiell zwei Möglichkeiten: Entweder verbindet man die Registereinheiten mit Ports und gibt über diese die Operanden und Befehle aus bzw. liest das Ergebnis ein, oder man baut sie als Speichereinheiten ein und spricht sie über den Daten- und Adressenbus direkt an.

Das Prinzip der zweiten Möglichkeit, die einfacher und bei der Verarbeitung schneller ist, soll hier kurz genauer erläutert werden. Als Beispiel dient der Fall, daß die Schaltung in einer 32-bit-Ausführung an einen Mikrocomputer mit 8-bit-Daten- und 16-bit-Adressenbus angeschlossen wird.

Die Verdrahtung für die Registereinheiten A und B ist einfach. Die Datenleitungen werden, vom Datenbus kommend, gepuffert und in vier parallel geführten Gruppen an die 32 Registereingänge gelegt. Die acht Wort-Takteingänge sind zu vier Paaren so zusammengeschaltet, daß bei einem Taktimpuls an eines der Paare entweder die ersten, die zweiten, die dritten oder die vierten 8 bit eingelesen werden. Ein 2-zu-4-Dekoder wählt aufgrund der 2 bit A0 und A1 des Adressenbus die entsprechende Taktleitung aus. Die anderen Adressenbit A2 ··· A15 werden in einem UND-Gatter so verknüpft, daß dieses nur bei einer bestimmten Konfiguration durchschaltet. Der Ausgang des UND-Gatters gibt mit einem H-Pegel den Dekoder frei. Dadurch können je acht bit des Registers genau unter einer Adresse angesprochen werden.

Die Auslesung des Registers E geschieht auf ähnliche Weise. Die Registerausgänge gelangen hier jedoch an die Eingänge eines acht bit breiten 4-zu-1-Multiplexers, der aus den vier Gruppen von je acht bit eine auf den Datenbus schaltet. Die Bit A0 und A1 bestimmen hier, welche der Gruppen ausgewählt wird, A2 ··· A15 werden, wiederum UND-verknüpft, an den Output-Control-Eingang geführt. Der Multiplexer muß unbedingt Tri-State-Ausgänge besitzen, die dann, wenn keine Auslesung erfolgt, hochohmigen Zustand annehmen.

Das Befehlsregister wird durch eine UND-Verknüpfung aller Adressenbit ebenfalls eindeutig adressiert.

Zu den Registern der Schaltung dürfen keine „normalen" Speichereinheiten parallel geschaltet sein; bei den Einheiten A und B wäre dies zwar theoretisch möglich, ist wegen der Gefahr eventuell auftretender Fehler nicht zu empfehlen.

## Hinweise zur Anwendung der Schaltung

Die Schaltung ist, wie schon mehrfach erwähnt, zur Verarbeitung von Gleitpunktzahlen gedacht. Es bestehen aber selbstverständlich auch weitere Anwendungsmöglichkeiten. Zu erwähnen ist hier besonders die Verwendung als n · n-Bit-Multiplizierer mit 2n-Bit-Ergebnis. Dazu müssen die Wort- und Erweiterungsteile beide gleich groß dimensioniert und beide ausgelesen werden. Damit läßt sich zusätzlich eine leistungsfähige Festkommaarithmetik realisieren.

Bei der Anwendung als Gleitpunktmultiplizierer oder -dividierer ist zu beachten, daß keine negativen Mantissen verarbeitet werden können. Vor der Operation müssen also eventuell einer oder beide Operanden negiert werden, was ebenfalls durch diese Schaltung geschehen kann. Bei der Division müssen sich außerdem die höchstwertigen gesetzten Bit der Operanden an der gleichen Stelle befinden, da sonst ein falsches Ergebnis geliefert wird. Dies muß gegebenenfalls vorher durch das Verschieben einer der Mantissen erreicht werden. Diese Verschiebung muß beim Ergebnis berücksichtigt und auf den Exponenten der Gleitpunktzahl umgerechnet werden.

Zum Schluß noch die Rechenzeiten: Eine 32-bit-Multiplikation/Division dauert mit einem 10-MHz-Takt ca. 6,4 µs, bei einem 20-MHz-Takt 3,2 µs.

## Befehlsworte für die verschiedenen Operationen

| | | | | M = H: logische Fkt. | M = L: Arithmetische Operationen | |
|---|---|---|---|---|---|---|
| | | | | $C_0$ = X | $C_0$ = H | $C_0$ = L |
| S3 | S2 | S1 | S0 | | | |
| L | L | L | L | $F = \overline{A}$ | F = A | F = A plus 1 |
| L | L | L | H | $F = \overline{A + B}$ | F = A + B | F = (A + B) plus 1 |
| L | L | H | L | $F = \overline{A}B$ | $F = A + \overline{B}$ | $F = (A + \overline{B})$ plus 1 |
| L | L | H | H | F = 0 | F = minus 1 | F = Null |
| L | H | L | L | $F = \overline{AB}$ | $F = A$ plus $A\overline{B}$ | F = A plus $A\overline{B}$ plus 1 |
| L | H | L | H | $F = \overline{B}$ | F = (A + B) plus $A\overline{B}$ | F = (A + B) plus $A\overline{B}$ plus 1 |
| L | H | H | L | $F = A \oplus B$ | F = A minus B minus 1 | F = A minus B |
| L | H | H | H | $F = A\overline{B}$ | $F = A\overline{B}$ minus 1 | $F = A\overline{B}$ |
| H | L | L | L | $F = \overline{A} + B$ | F = A plus AB | F = A plus AB plus 1 |
| H | L | L | H | $F = \overline{A \oplus B}$ | F = A plus B | F = A plus B plus 1 |
| H | L | H | L | F = B | $F = (A + \overline{B})$ plus AB | $F = (A + \overline{B})$ plus AB plus 1 |
| H | L | H | H | F = AB | F = AB minus 1 | F = AB |
| H | H | L | L | F = 1 | F = A plus A | F = A plus A plus 1 |
| H | H | L | H | $F = A + \overline{B}$ | F = (A + B) plus A | F = (A + B) plus A plus 1 |
| H | H | H | L | F = A + B | $F = (A + \overline{B})$ plus A | $F = (A + \overline{B})$ plus A plus 1 |
| H | H | H | H | F = A | F = A minus 1 | F = A |

Peter Rödler

# Softkeys für CBM

Die Möglichkeit des Einbaus von „Softkeys“ in den CBM bzw. jedes anderen Computers mit einem User-Port wie bei Commodore erlaubt die Erstellung von sehr komfortablen Menüprogrammen, bei denen Fehlbedienung zudem weitestgehend ausgeschlossen ist.

In der einfachsten Version besteht die Schaltung aus vier Schaltern, die auf eine Lochplatte gelötet werden. Diese Platte ist in der Breite so bemessen, daß sie in den Spalt unter dem Bildschirm des CBM gepreßt werden kann.

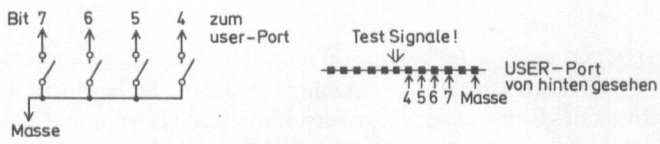
Alle Taster werden auf Masse gelegt und schalten diese auf die oberen 4 bit des User-Port. (Diese wurden gewählt, da ein eigenes, aber auch viele andere Schaltinterfaces das untere Halbbyte benützen.) Der Anschluß kann mit Hilfe eines entsprechenden Steckers am Port erfolgen. Ich habe, um den Port für das Schaltinterface freizubehalten, die Verbindungsdrähte direkt auf der Platine aufgelötet. Diese mußte dazu jedoch aus dem Computer ausgebaut werden, da sich die Anschlüsse auf der Unterseite befinden. Achtung! Nicht aus Versehen die Oberseite benützen.

Die Abfrage der Schalter wird von dem Beispielprogramm 1 geleistet. Vorher muß mit zwei Printbefehlen die Funktion der Schalter der Bedienung des Programmes mitgeteilt werden.

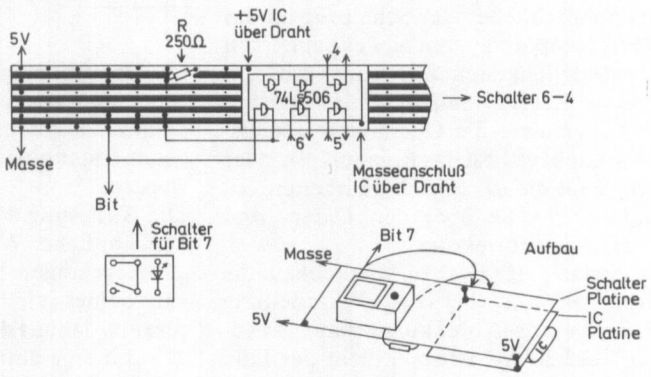
In der erweiterten Version der Schaltung werden statt der normalen Taster solche mit eingebauten Leuchtdioden benutzt. So wird es möglich, die jeweils aktiven Schalter nochmals optisch hervorzuheben. Diese Version ist also mehr oder weniger nur um eine Spielerei erweitert und ist nicht unbedingt besser. Hierzu wird jedoch zum einen ein etwas höherer Aufwand bei der Software, zum anderen der Einsatz eines Treiber-IC 7406 zur Ansteuerung der Dioden nötig. Invertierende Treiber wurden deshalb benutzt, damit die Dioden nicht immer leuchten, sondern nur im Anwendungsfall.

Die Schalter werden wie in der ersten Version geschaltet. Von ihrem Ausgang geht jedoch ein Draht zum Eingang eines Inverters des 7406 und ein weiterer von dem entsprechenden Ausgang zurück zum Minusanschluß der eingebauten Leuchtdiode. Die Verbindung des Pluspols der Leuchtdiode mit der Betriebsspannung geschieht wegen der Strombegrenzung über einen 200- bis 250-Ω-Widerstand.

Die Kabel der so hergestellten Platine werden durch ein Loch im Gehäuse des PET nach innen geführt. Der Anschluß ist einfach. Das zusätzliche 5-V-Kabel wird an den 5-V-Ausgang für den Recorder 2 angeschlossen. Die ganze Platine muß, da das Gehäuse des CBM aus Me-




**Schalterstellung zum USER-Port und Anschlußschema für CBM-Computer**




**Integrierter Baustein 74LS06 auf der Platine mit Schalter für Bit 7 und dem Aufbau**

**Schalterplatine für die Softkeys**

**Mondlandung mit „Handsteuer“**

tall ist, mit Isolierband auf der Rückseite gut gegen Kurzschluß gesichert sein.

Die Abfrage der Schalter und die Ansteuerung der ausgewählten Leuchtdioden leistet Programm 2.

Hierzu noch einige Anmerkungen:
Durch den Rücksprung in Zeile 4 und 5 nach Zeile 1 werden Fehlbedienungen ignoriert. (Da die Dauer der Durchläufe durch dieses Programmteil sich dabei ändert, verändern die aktiven Leuchtdioden, wie um auf den Fehler hinzuweisen, ihre Blinkfrequenz. – Zufall, nicht beabsichtigt, aber auch ganz nett.)

Der Wert des ersten Pokebefehls ergibt sich aus der Addition der binären Werte der Schalter bzw. der von ihnen bedienten Eingänge am User-Port und – in meinem Fall, um die unteren Leitungen als Ausgang dem Schaltinterface zur Verfügung zu stellen, was auch unterbleiben kann – dem Wert 15.

**Tabelle 1**

1. Programm ohne Leuchtdioden, Schalter 1–4 aktiv

```
  2 FORI=4TO7:IFNOTPEEK(59457)AND2↑ITHEN4
  3 NEXTI
  4 ON8-IGOTO100,200,300,400
  5 GOTO2
100 PRINT1:GOTO2
200 PRINT2:GOTO2
300 PRINT3:GOTO2
400 PRINT4:GOTO2
READY.
```

2. Programm mit Leuchtdioden, Schalter 1 + 3 aktiv

```
  1 POKE59459,175:POKE59459,15:REM 15(unteres
  Halbbyte)+128(S1)+32(S2)=175
  2 FORI=4TO7:IFNOTPEEK(59457)AND2↑ITHEN4
  3 NEXTI
  4 ON8-IGOTO100,1,300,1
  5 GOTO1
100 PRINT1:GOTO1
300 PRINT3:GOTO1
READY.
```

Andreas Dworschak

# Joystick für Mikroprozessoren

Sinn dieser Anleitung ist es, möglichst einfach und billig mehrere beliebige Joysticks an einen Mikrocomputer anzuschließen. Dabei sollte gleichzeitig die Möglichkeit gegeben sein, später noch mehr Joysticks anzuschließen bzw. die Auflösung zu erhöhen.

## Das Funktionsprinzip

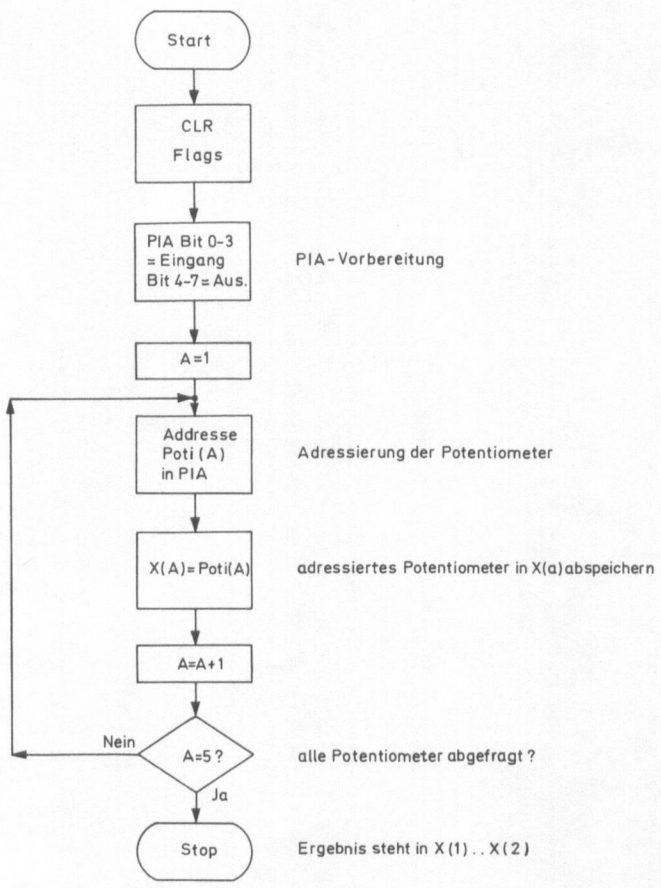
Ein Joystick (z.B. aus einem Fernsehspiel) besteht aus der Hebelmechanik und zwei Potentiometern, wobei ein Potentiometer für die X-Steuereinrichtung und das andere für die Y-Steuereinrichtung gedacht ist. Nun gilt es, lediglich die an den Potentiometern abgegriffenen Spannungen dem Mikroprozessor verständlich zu machen. Dies geschieht mit einem Analog/Digital-Wandler.
Um die Kosten für den A/D-Wandler möglichst gering zu halten, wurde ein LED-Ansteuerbaustein UAA 170 benutzt. Bei zwei Joysticks müssen vom Prozessor bereits vier Potentiometer abgefragt werden. Um dies zu ermöglichen, müssen die vier Potentiometer nacheinander auf den Steuereingang des UAA 170 gegeben werden, was mit Hilfe eines analogen Multiplexers geschieht.
Da die Ausgänge des UAA 170 keine TTL-Pegel liefern, müssen sie über die Widerstände R1 bis R4 und die Z-Dioden Z1 bis Z4 an den 74LS00 angepaßt werden.

## Die Auflösung der Joysticks

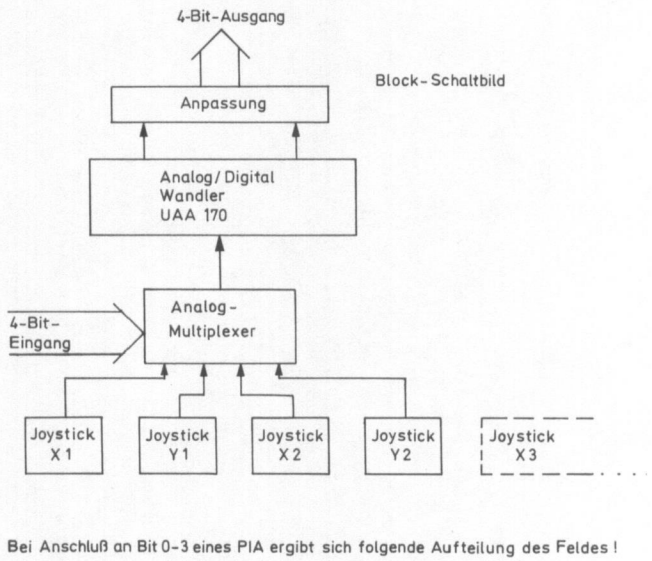
Beim UAA 170 werden nur die Ausgänge 2 bis 5 benutzt. Der vom Steuerhebel zurückgelegte Weg wird dadurch in vier Teile geteilt, wobei jeweils einer von den Ausgängen 2 bis 5 durchgeschaltet wird. Da jedes Potentiometer so geteilt wird, erreicht man pro Joystick eine Auflösung von $4 \times 4 = 16$; d.h., der Prozessor kann 16 verschiedene Stellungen des Steuerhebels erkennen, was im Normalfall ausreichend ist. Welches Bit (0 bis 3) bei der jeweiligen Stellung „H" ist, ist der beigefügten Tabelle zu entnehmen.
Wenn die Ausgänge 6 bis 9 des UAA 170 noch dazugenommen werden, kann die Auflösung auf $16 \times 16 = 256$ gesteigert werden. In diesem Fall sind natürlich acht PIA-Eingänge erforderlich.

## Die Schaltung

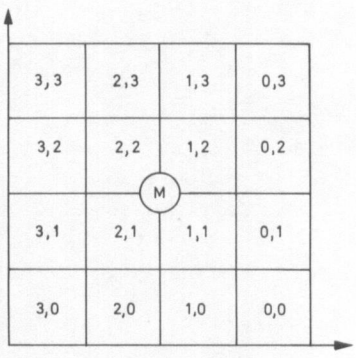
Der analoge Multiplexer wird mit einem Baustein vom Typ 4016 aufgebaut. Dieser Baustein enthält vier Analogschalter, die über vier Bit eines PIA direkt angewählt werden können. Der Einsteller R14 wird benötigt, um Anfangs- und

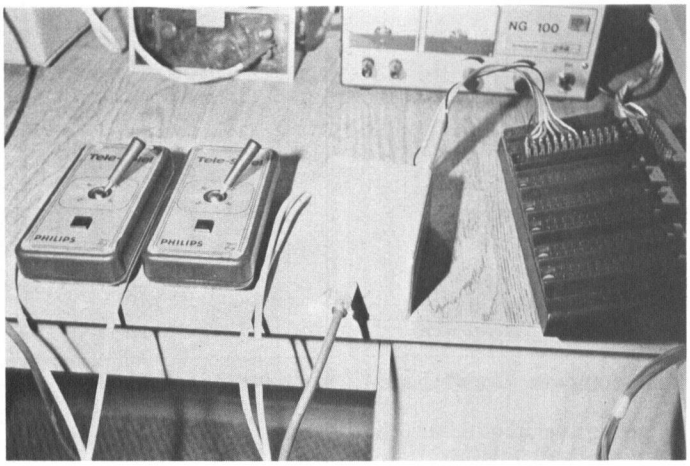



**Flußdiagramm für die Joystick-Schaltung**



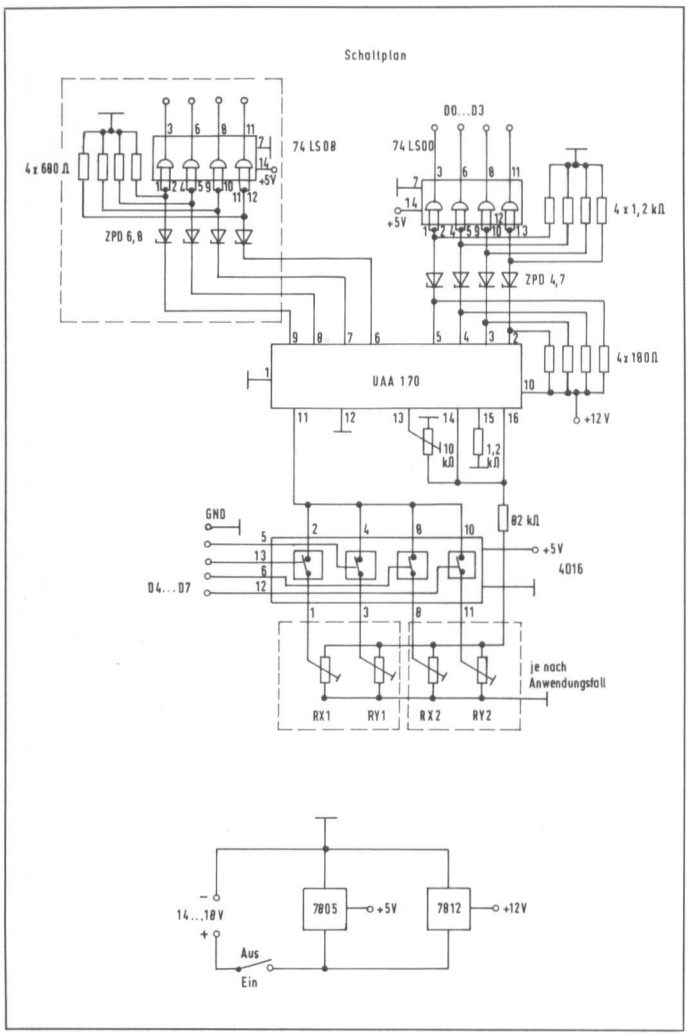

Bei Anschluß an Bit 0-3 eines PIA ergibt sich folgende Aufteilung des Feldes !



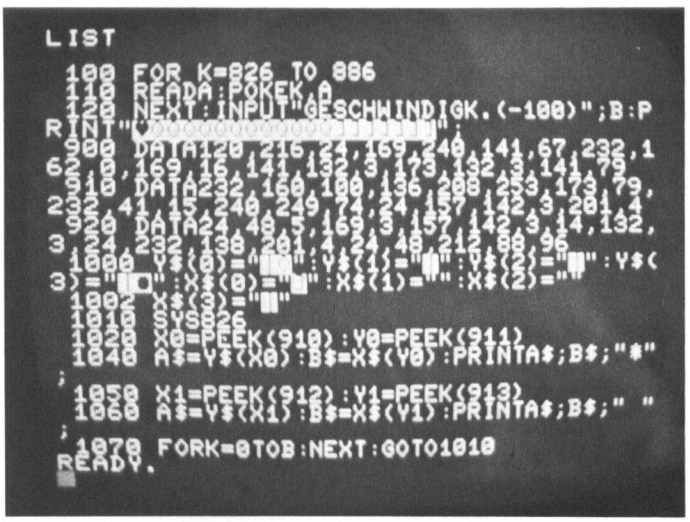



**Joysticks am Computer**

**Computeranlage des Autors**




**Schaltung mit dem Baustein UAA 170 für die Joysticks**



```
LIST
 100 FOR K=826 TO 886
 110 READA:POKEK,A
 120 NEXT:INPUT"GESCHWINDIGK.(-100)";B:P
RINT"
 900 DATA120,216,24,169,240,141,67,232,1
62,0,169,16,141,132,3,173,132,3,141,79,
 910 DATA232,160,100,136,208,253,173,79,
232,41,15,240,249,24,24,157,142,3,201,4,
 920 DATA24,40,5,169,3,157,142,3,14,132,
3,24,232,138,201,4,24,48,212,88,96
 1000 Y$(0)="":Y$(1)="":Y$(2)="":Y$(
3)="":X$(0)="":X$(1)="":X$(2)=""
 1002 X$(3)=""
 1010 SYS826
 1020 X0=PEEK(910):Y0=PEEK(911)
 1040 A$=Y$(X0):B$=X$(Y0):PRINTA$;B$;"#"
;
 1050 X1=PEEK(912):Y1=PEEK(913)
 1060 A$=Y$(X1):B$=X$(Y1):PRINTA$;B$;" "
;
 1070 FORK=0TOB:NEXT:GOTO1010
READY.
```

**Bildschirmausgabe der Software**

## Der Aufbau

Der Aufbau bringt keinerlei Probleme mit sich und kann auf einer Experimentierplatine vorgenommen werden. Wer allerdings für saubere Arbeit ist, kann die Schaltung auch auf einer selbstgemachten Platine unterbringen.

Der Anschluß an den Prozessor erfolgt über 8 bit eines I/O-Bausteines, wobei vier als Eingang (0 bis 3) und vier als Ausgang (4 bis 7) geschaltet werden müssen. Die vier Ausgänge dienen zum Ansteuern des 4016.

## Die Software

Anhand des Flußdiagrammes kann die Software leicht erstellt werden.

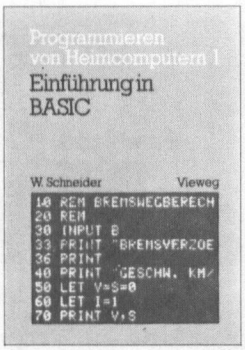
## Die Erweiterung

Wem zwei Joysticks zuwenig sind, der kann den Analogmultiplexer durch Hinzufügen weiterer Bausteine vom Typ CM4016 erweitern.

Endpunkt des Ausschlages festzulegen. R14 muß nach dem Zusammenbau dementsprechend eingestellt werden.

Die Joystick-Einsteller können Werte von etwa 10 bis 500 kΩ annehmen, wobei die für die X-Richtung und für die Y-Richtung nicht unbedingt gleiche Werte haben müssen. Als Betriebsspannung werden +5 V und +12 V benötigt. Bei Benutzung von Spannungsreglerbausteinen des Typs 7805 und 7812 kann eine einzige Betriebsspannung im Bereich von 14 V bis 18 V benutzt werden.

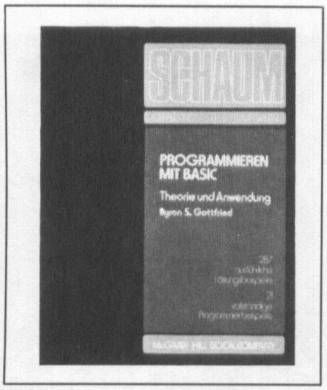
Als Spannungsquelle für die Joystick-Potentiometer wird die Spannungsquelle des UAA 170 (Pin 14) benutzt (5,1 V).




**Programmieren von Heimcomputern 1 Einführung in BASIC**

Schneider, Wolfgang, 148 Seiten, 24 DM

In den einzelnen Kapiteln wird besonderer Wert auf einprägsame Merksätze und lehrreiche Beispiele gelegt. Am Ende eines jeden Kapitels ist das Wichtigste zum Nachschlagen und zum Wiederholen konzentriert wiedergegeben. Vollständig programmierte Beispiele zeigen darüber hinaus, wie man eigene Programme entwickelt.




**Programmieren mit Basic**

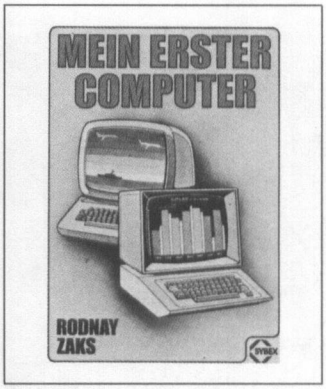
Gottfried, Byron, S., 238 Seiten, 29,80 DM

Der im ständigen Wechsel zwischen Theorie und praktischen Beispielen leicht verständlich aufbereitete Stoff führt in acht Kapiteln von den Grundlagen bis zu Vektor und Matrizen-Operationen. Schritt für Schritt wird die Entstehung des Programms erklärt, dazu das Flußdiagramm. Mathematisch und auch physikalisch Interessierte kommen ebenso auf ihre Kosten wie Spielprogrammfans.

**Mikrocomputer-Grundwissen**

Allgemeinverständliche Einführung in die Mikrocomputer-Technik
Osborne, Adam, 300 Seiten, 36 DM

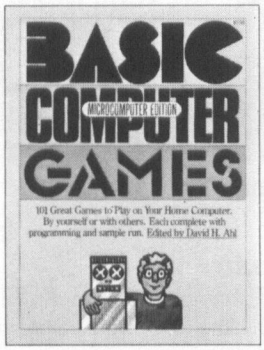
Deutsche Erstauflage eines Osborne-Bestsellers für Elektronikneulinge. Ohne großen Zeit- und Kostenaufwand kann im Selbststudium das Grundwissen erarbeitet werden. Nach sechs Lernschritten beherrscht der Leser die Materie und kann mitreden, wenn es um den Mikrocomputer geht.




**Mein erster Computer**

Rodnay Zaks, 304 Seiten, 28 DM

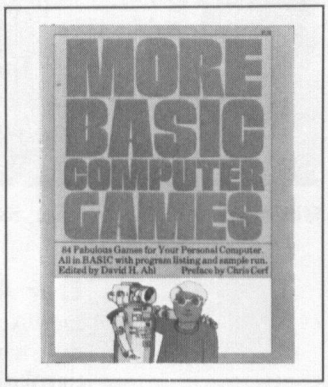
„Mein erster Computer" ist die Übersetzung von „Your first Computer". Es geht darum, was ein Mikrocomputer ist, wie er funktioniert und wofür er zu gebrauchen ist. Nach der Lektüre des Buches sind Sie selbst in der Lage zu entscheiden, ob ein Computer für Sie nützlich ist, welchen Rechner und welche Zubehörgeräte Sie gegebenenfalls brauchen. Sie müssen weder Techniker sein noch Elektronikvorkenntnisse haben.




**Basic Computer Games**

Ahl, David H., 185 Seiten, 38 DM (in Englisch)

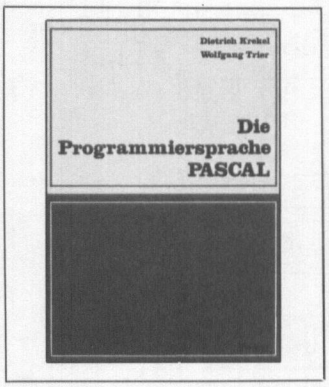
Es darf gespielt werden – mit Basicprogrammen. Dieses Buch in englischer Sprache ist ein Dorado für Spielprogrammfans. 101 Spiele (mit Erklärung und Programmliste) verschiedener Art sorgen für Freude aller Benutzer. Jede Kategorie ist mit einer größeren Zahl vertreten: Weltraum (Super Star Trek), Sport (Bowling, Hockey), Logikspiel (Digits, Word, Tic Tac Toe), Casinospiele (Poker, Roulette), Bildungsspiele (Change). Bei diesem Buch steht der Lerneffekt nicht im Vordergrund.




**More Basic Computer Games**

Ahl, David H., 188 Seiten, 38 DM (in Englisch)

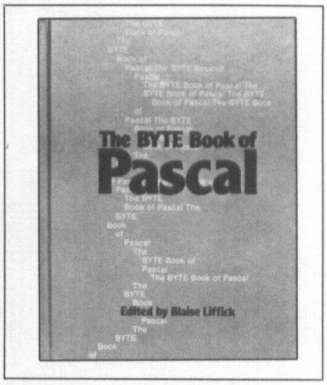
Computerspiele gewinnen immer mehr Freunde. Dieser Folgeband bietet wieder eine Menge Spaß und Anregung mit 84 brandneuen Spielen, ausgestattet mit den fantasievollsten Attributen: Mit „Camel" lernen Sie das Überleben in der Wüste, „Father" diskutiert an Ihrer Stelle mit den Kindern, die einfach nicht ins Bett gehen wollen, und durch „Millionaire" erhalten Sie Gelegenheit, Ihr Leben noch einmal zu planen.




**Die Programmiersprache PASCAL**

Dietrich Krekel, Wolfgang Trier, 587 Seiten, 68 DM

PASCAL ist eine kleine, leicht erlernbare, maschinentypunabhängige Programmiersprache. Sie ermöglicht problemnahe, gut strukturierte und effiziente Programme. Der Trend zur strukturierten Programmierung und die Mikrocomputer haben die Verbreitung dieser Sprache begünstigt. Das vorliegende Buch stellt, dem Aufbau der Sprache folgend, Symbole, Datentypen, Variable, Ausdrücke, Anweisungen und Unterprogramme exakt dar. Wichtige Besonderheiten sind in Regeln zusammengefaßt.

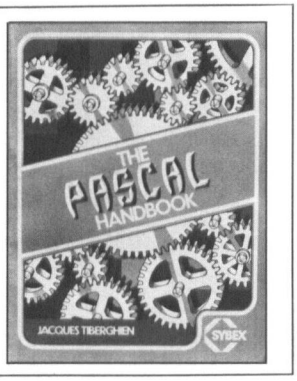



**The BYTE Book of PASCAL**

Blaise W. Liffick, 333 Seiten, 80 DM (engl.)

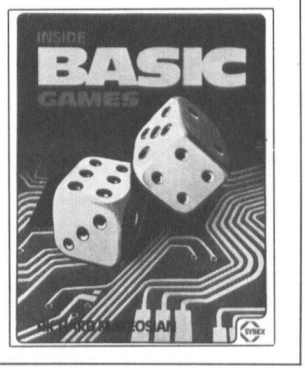
Das Buch aus der BYTE-Serie „Sprachen" enthält Aufsätze der Zeitschrift BYTE. Damit ist so etwas wie eine PASCAL-Einführung entstanden. Die Möglichkeiten und Vorzüge der Sprache werden diskutiert und mit BASIC und COBOL verglichen. Der letzte Teil des Buches ist der Anwendung gewidmet. Dazu gehören System- und Anwender-Programme in PASCAL. So ist die Sammlung auch eine Fundgrube für entsprechende Software.

**Alle Bücher erhalten Sie bei Ihrem Buchhändler**



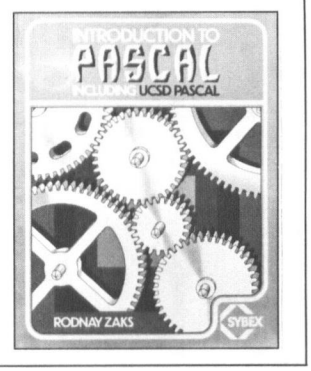

**The PASCAL Handbook**
Jacques Tiberghien, 471 Seiten, 54 DM (engl.).

Das PASCAL-Handbuch ist etwas mehr als eine kommentierte Liste der PASCAL-Befehle. Das Handbuch ist angelegt als umfassendes Nachschlagewerk für die verschiedenen PASCAL-Dialekte – mit dem Versuch, sie mit dem neu definierten ISO-Standard (International Standards Organisation) in Einklang zu bringen. Das Buch ist deshalb von unschätzbarem Wert gerade für den professionellen Programmierer, der für unterschiedliche Rechner Programme schreibt.




**Inside BASIC Games**
Richard Mateosian, 325 Seiten, 54 DM

„Inside BASIC Games" ist mehr als eine bloße Sammlung von Computer-Spielen. Mateosian will dem Leser helfen, eigene Spielprogramme zu entwickeln und dabei natürlich die BASIC-Kenntnisse zu vervollkommnen. Doch auch wer nur spielen will, kommt auf seine Kosten: Arithmetic Drill, Clock, Card Memory, Ten-Key Flicker, Timer, Birthday, Match-Up oder Alien Encounter sind die Namen von Spielen, die der Autor selbst entworfen hat.

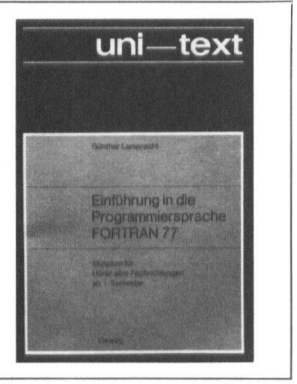



**Introduction to PASCAL**
Rodnay Zaks, 422 Seiten, 57 DM

Das richtige Buch für jeden, der PASCAL lernen will. Programmierkenntnisse setzt Rodnay Zaks beim Leser nicht voraus. Sein einfaches und doch umfassendes Werk führt schrittweise in das Wesentliche von Standard- und UCSD/PASCAL ein. Abgestufte Übungen vertiefen das Erlernte so, daß bald das Erstellen eigener Programme möglich ist. Genau dies, Lernen durch Programmieren, legt der Autor seinen Lesern besonders ans Herz.

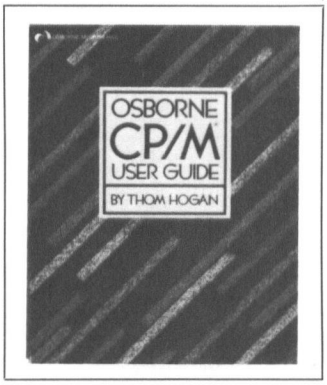
**Mathematische Unterhaltungen und Spiele**
Hans Heinrich Gloistehn, 204 Seiten, 24,80 DM

Das Buch gibt eine Auswahl mathematischer Spielereien, die sich gut für einen programmierbaren Taschenrechner eignen. Es wendet sich an den interessierten Laien, für den Mathematik vor allen Dingen ein Hobby ist. Grundbegriffe des Programmierens werden vorausgesetzt, große mathematische Vorkenntnisse sind nicht erforderlich. Die einzelnen Abschnitte sind so abgefaßt, daß sie jeweils für sich lesbar sind.



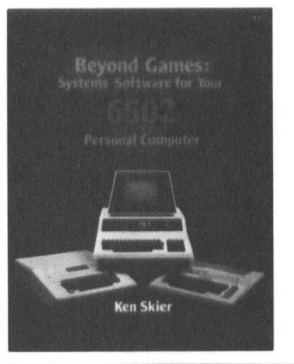

**Einführung in die Programmiersprache FORTRAN 77**
Günther Lamprecht, 136 Seiten, 19,80 DM.

FORTRAN IV ist die im wissenschaftlichen Bereich am weitesten verbreitete Programmiersprache. Obwohl FORTRAN IV als höhere Programmiersprache rechnerunabhängig sein sollte, hat die Vielzahl rechner- und anwendungsspezifischer FORTRAN-Dialekte einen neuen Standard nötig gemacht. Dies ist mit FORTRAN 77 geschehen. Lamprecht führt mit seinem Buch anhand von Beispielen und Aufgaben in diese Programmiersprache ein, wobei er auf Abweichungen von FORTRAN IV eingeht.



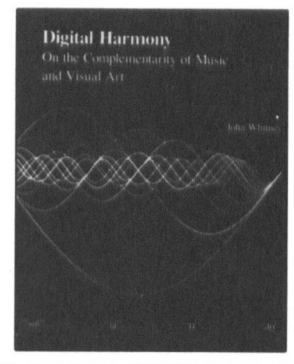

**CP/M User Guide**
Thom Hogan, 283 Seiten, 43 DM (engl.).

CP/M gilt heute als Standardbetriebssystem für Mikrocomputer. Es erfordert keine große Computererfahrung. Das Buch macht daher den Neuling zugleich mit dem Mikrocomputer vertraut. Das beginnt mit praktischen Informationen, um starten zu können. Dann werden die CP/M-Befehle näher erklärt, das tägliche Brot im Umgang mit dem Gerät. Beispiele schaffen ein dauerhaftes Fundament zum Verständnis des Betriebssystems.




**Beyond Games: Systems Software for Your 6502 Personal Computer**
Ken Skier, 443 Seiten, 50 DM (engl.)

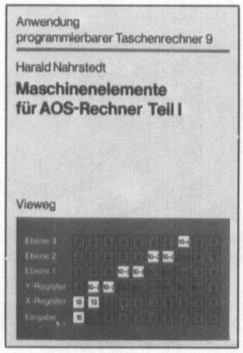
Mit seinem Buch will Ken Skier alle Besitzer von Personal Computern auf der Basis des Mikroprozessors 6502 ermuntern, ihr Gerät für mehr zu nutzen als nur zum Spielen. Wer also beispielsweise einen Apple-, Atari- oder Commodore-Rechner sein eigen nennt, erhält hier eine gründliche Einführung. Dazu gehört auch eine kurze, aber erstaunlich vollständige Behandlung des Programmierens in Assembler.




**Digital Harmony – On the Complementary of Music and Visual Art**
John Whitney, 235 Seiten, 72 DM (engl.)

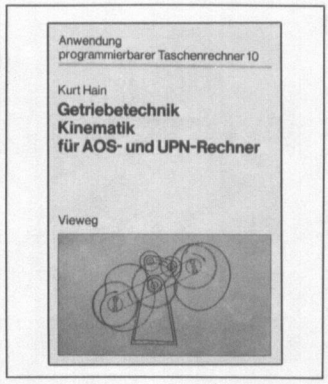
Die Computer verwirklichen einen alten Traum der Künstler: Bilder zu schaffen, die so aussehen, wie Musik klingt. Whitney schildert hier, wie seine Grafiken entstanden sind. Er beschreibt seinen Computer, Peripherie, Filmtechnik, seine Programme, kurz alles, was zum Einstieg in diesen technisierten Zweig der Kunst nötig ist. Keinen Zweifel läßt er daran, daß er sich als Künstler, weniger als Programmierer versteht.

**Alle Bücher erhalten Sie bei Ihrem Buchhändler**



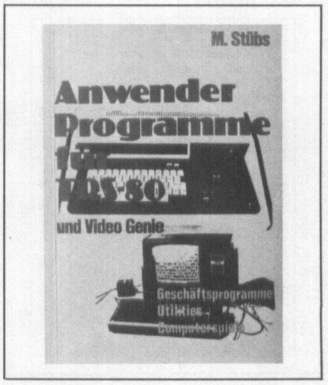

**Maschinenelemente für AOS-Rechner Teil I: Grundlagen, Verbindungselemente, Rotationselemente**
Harald Nahrstedt, 171 Seiten, 34 DM

Mit Programmen zur Berechnung von Schrauben und Federn sowie zur Bestimmung der Durchbiegung von Achsen und Wellen legt Nahrstedt Grundsteine zu rationellem Arbeiten in der Konstruktion. Langwierige Rechenverfahren werden überflüssig, und konstruktive Detailänderungen können ohne großen Aufwand realisiert und optimiert werden.




**Getriebetechnik, Kinematik für AOS- und UPN-Rechner**
Kurt Hain, 102 Seiten, 38 DM

Konstrukteuren von Maschinen mit ungleichförmig übersetzenden Getrieben bieten programmierbare Taschenrechner weitreichende Möglichkeiten. Jetzt lassen sich nicht nur die Konstruktionen in ihrer Auslegung, sondern auch die getriebetechnischen Berechnungsverfahren optimieren. Das Buch von Hain stellt dafür Programme zur Verfügung. Was Studenten begrüßen werden: Kenntnisse der Getriebetechnik sind nicht vorausgesetzt.




**Anwenderprogramme für TRS-80 und Video Genie**
Martin Stübs, 165 Seiten, 29,80 DM

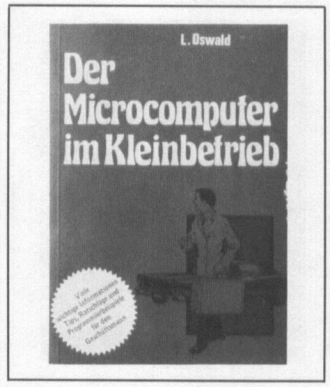
Hier geht es nicht um theoretische Abhandlungen, sondern um Programme – um Programme aus den beiden großen Bereichen Computerspiele und berufliche Anwendung. Der Leser kann, sollte aber nicht, die Programme nur einfach benutzen. Denn anhand der Programme läßt sich auch lernen, auf welchen Wegen typische Programmierprobleme zu lösen sind. Die Beispiele im Buch sind teils in Level II-BASIC, teils in Disk-BASIC geschrieben.

**16 Bit Microcomputer – Einführung, Daten, Eigenschaften, Anwendung**
Josef Koller, 374 Seiten, 29,80 DM

Das Buch beschäftigt sich mit 16-bit-Mikroprozessoren, und zwar mit den Typen 9900 (Texas Instruments), 8086 (Intel), Z8000 (Zilog) und 68 000 (Motorola). Wie die Prozessoren und dazugehörige Peripheriebausteine arbeiten, ist eingehend dargestellt. Den Schluß des Buches bestreiten Schaltpläne für einfache Rechnersysteme der behandelten Prozessoren.

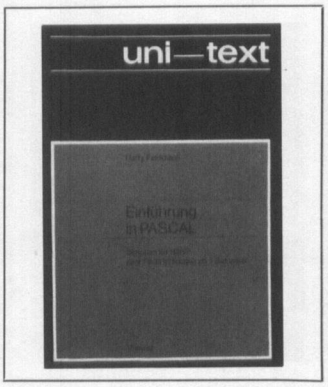
**Programme für CBM**
L. Oswald, 119 Seiten, 19,80 DM

Die „Programme für CBM" sollen PET/CBM-Besitzern die Möglichkeit geben, ihr System mit geringem Aufwand mit leistungsfähigen Programmen füttern zu können. Bei der Zusammenstellung der Programme hat Oswald sowohl an den professionellen Anwender als auch den Hobbyisten gedacht. Deshalb finden sich neben Fakturierprogrammen und Textverarbeitung auch Spiele. Die Programme verwenden kaum PEEK- und POKE-Befehle, so daß die meisten Programme auf allen CBM- und PET-Computern laufen dürften.



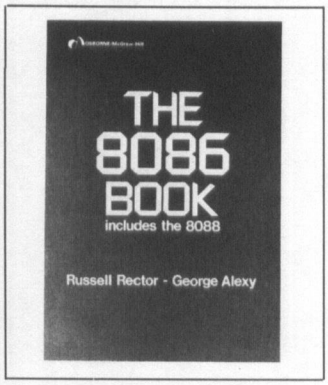

**Der Microcomputer im Kleinbetrieb**
L. Oswald, 174 Seiten, 39,80 DM

Wie setzt man Mikrocomputer im Betrieb ein? Mit seinem Buch will Oswald solchen Geschäftsleuten den Einstieg in das Gebiet der Mikrocomputer ebnen, die noch keine nennenswerte Erfahrung im Umgang mit der elektronischen Datenverarbeitung haben. Oswald beginnt zwar bei „Adam und Eva", bringt es aber trotzdem fertig, unter Umgehung der entwicklungsgeschichtlich interessanten Phase der Computer-Spiele direkt aufs Wesentliche zu kommen: Inventur, Adressen, Buchhaltung usw.




**Einführung in PASCAL**
Harry Feldmann, 136 Seiten, 19,80 DM.

Unter den modernen universellen, für strukturiertes Programmieren geeigneten Sprachen ist PASCAL am wenigsten umfangreich. Feldmann beschreibt den vollen PASCAL-Standard und behandelt die wichtigsten Erweiterungen. Die systematische Darstellung macht sein Buch zu einem PASCAL-Nachschlagewerk, das durch interessante Beispiele motiviert, das durch die Vollständigkeit der grammatischen Regeln einschließlich des Syntaxschemas und der Auflistung aller Standardvereinbarungen umfassend informiert.




**The 8086 Book includes the 8088**
Russell Rector, George Alexy, 606 Seiten, 54,– DM (engl.)

Das 8086-Buch ist keine Einführung in die Mikroprozessortechnik, es setzt vielmehr solche Kenntnisse voraus – ist dabei aber ein verständliches und vollständiges Nachschlagewerk für die Eigenschaften des 8086. Die Architektur des Prozessors ist erklärt, der ganze Befehlssatz ist aufgeführt mit einer Anleitung zur optimalen Programmierung. Die Schnittstellentechnik wird diskutiert. Kurz, das Buch ist eine wichtige Hilfe für jeden, der einen Mikrocomputer auf der Basis des 8086 benutzen will.

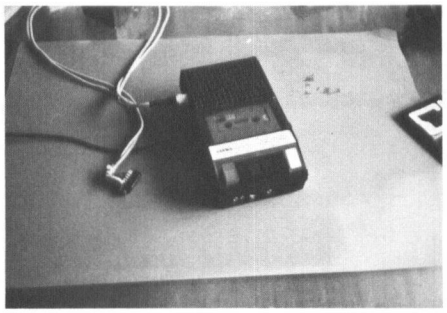
**Alle Bücher erhalten Sie bei Ihrem Buchhändler**

Jens Schultheiß

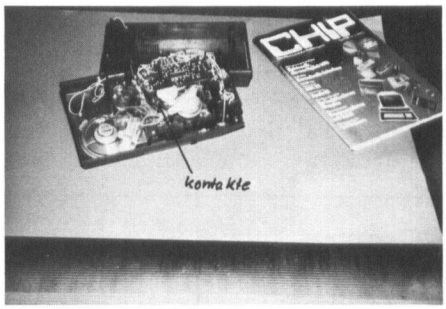
# Anpassung eines Kassettenrecorders an einen CBM

Ein handelsüblicher Kassettenrecorder wurde so modifiziert, daß er die Aufgabe des Recorders von CBM übernehmen kann. Gewählt wurde der Typ 2033A von SANWA.
In den Recorder wurde eine sechspolige Buchse eingebaut, über deren Kontakte folgende Signale geführt wurden:

- Masse,
- Eingang,
- Ausgang,
- Motor,
- Kassettenschalter.

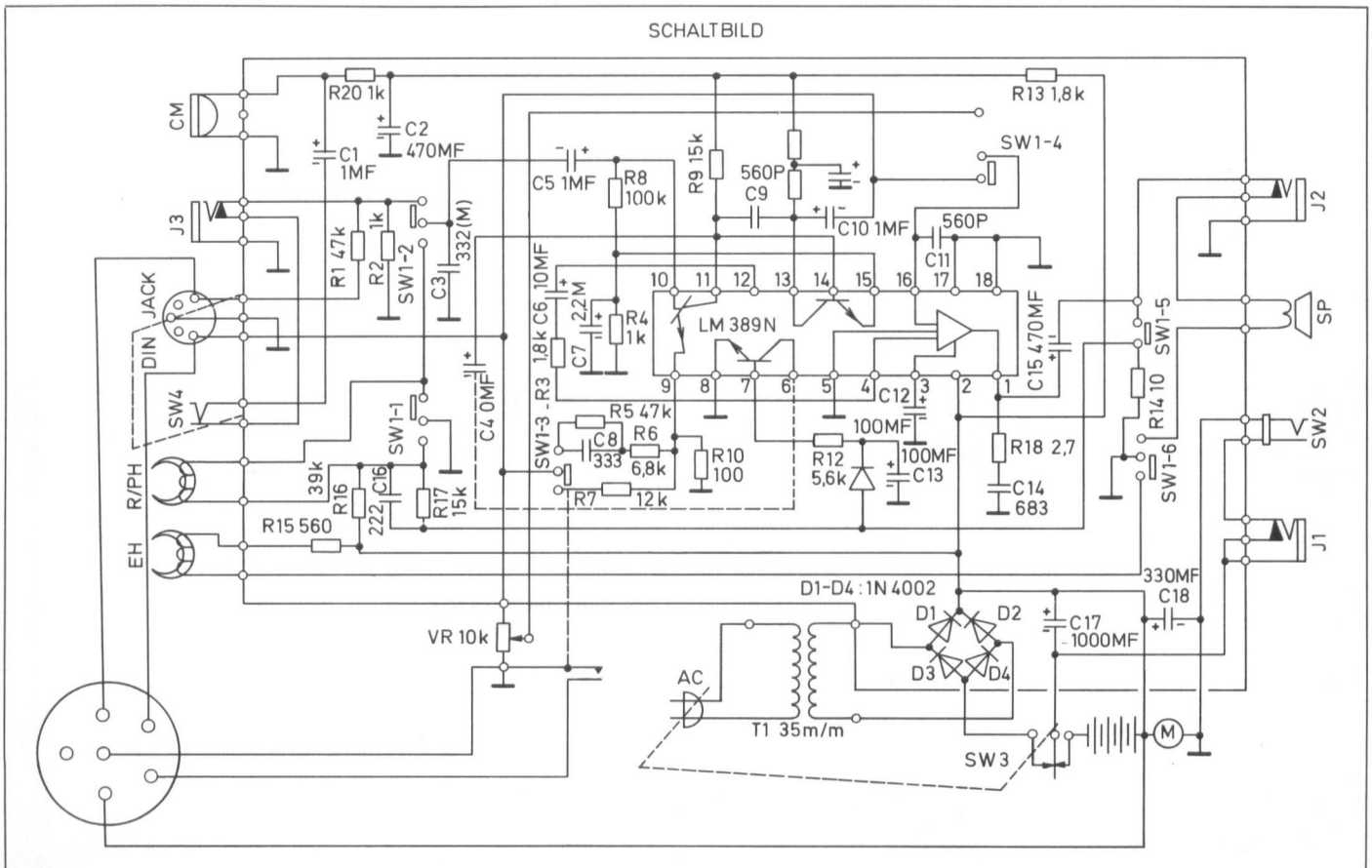


**Kassettenrecorder für CBM-Computer als Interface-Erweiterung**




**Geöffneter Kassettenrecorder mit Bezeichnungen für den Umbau**

Eingang und Ausgang wurden mit den entsprechenden Kontakten der Audiobuchse verbunden. Ein zusätzlicher Kontakt, der bei „play" oder „record" betätigt wird, wurde ebenfalls eingesetzt. Er legt das Signal „cross switch" an Masse. Die automatische Aufnahmekontrolle wurde unterbrochen. Der Lautstärkeregler steht auf Maximum, und der Lautsprecher wurde durch einen Vorwiderstand von 120 Ω auf Hörlautstärke reduziert. Dadurch ist eine akustische Kontrolle des Ladevorganges möglich. Der Lautsprecher kann aber auch abgeschaltet werden.
Die Stromversorgung des Motors geschieht aus dem Rechner. Der Verstärker wird über die eingebaute Spannungsversorgung des Recorders betrieben.




**Schaltung des umgebauten Kassettenrecorders für CBM-Computer**

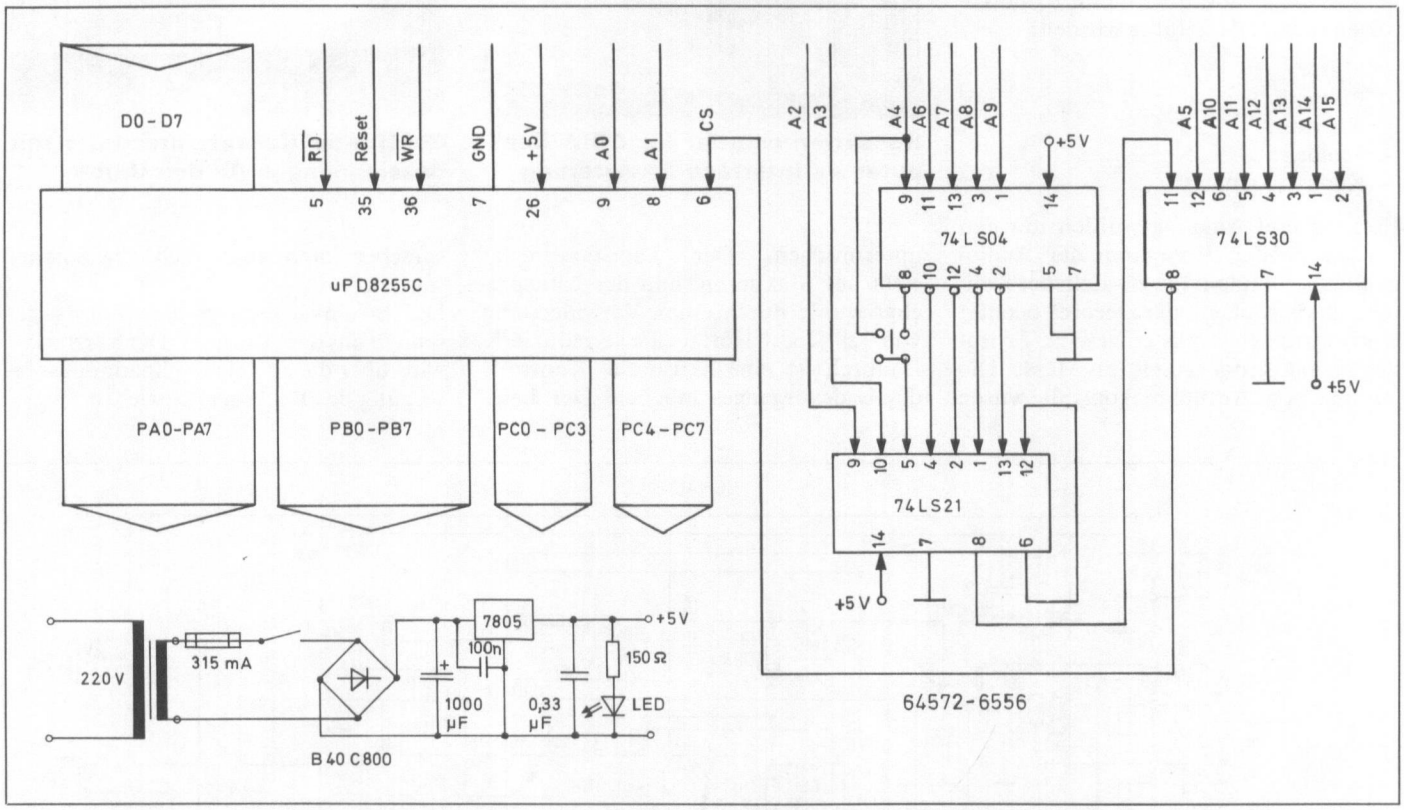
Herbert Draht

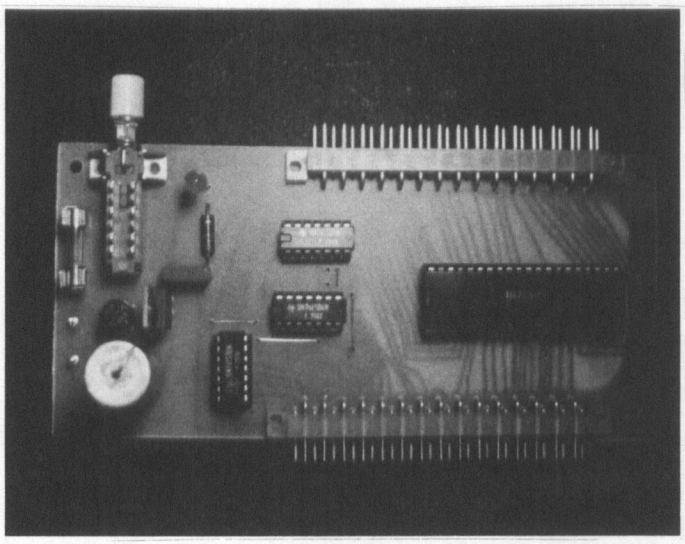
# 3 × 8-bit-Parallelschnittstelle für MZ-80K

Den bei weitem breitesten Bereich innerhalb der Interface-Baugruppen deckt diese Input/Output-Karte ab. Sie ist mit einem PIO-Baustein 8255 ausgestattet und hat damit drei 8-bit-Kanäle. Jeder Kanal kann als Byte-Ein-/Ausgabe arbeiten; der Port C kann dabei noch in zwei Nibble (Nibble = 4-bit-Wort) unterteilt werden. Die „zwei C-Porten" können auch unterschiedlich programmiert werden.

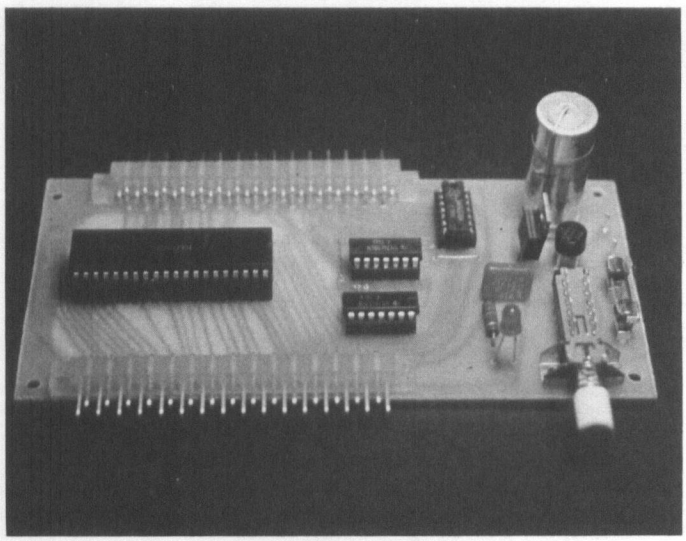
Die 3 × 8-bit-Schnittstelle kann vielfältig verwendet werden. Hier einige Beispiele:

- Frequenz- und Periodendauermessungen,
- Logikanalysator,
- EPROM-Programmierer,
- D/A-Wandler.

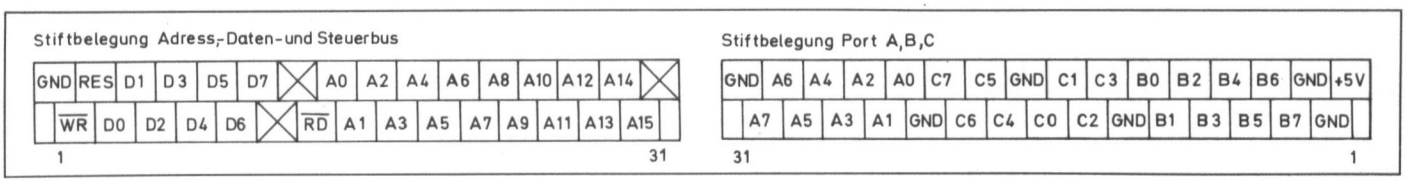


**Aufbau für die 3 × 8-bit-Parallelschnittstelle**



**Parallelschnittstelle für den MZ-80K**



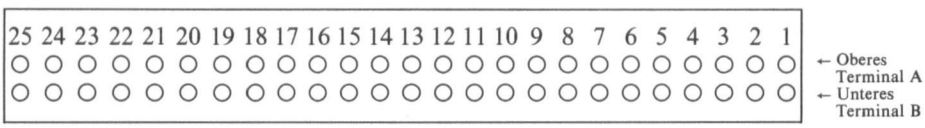

Der Baustein 8255 läßt auch Quittungsbetrieb mit Druckern oder ähnlichen peripheren Schaltungen zu.
Die Ein-/Ausgänge der PIO sind auf einer 31pol. Leiste herausgeführt. Die Verbindung mit Ihrem Personalcomputer erfolgt über den Busstecker. Die Stiftbelegung des Bussteckers und des Interfacesteckers sind angegeben. Falls Sie mit der Zählweise des MZ-80K-Steckers in Schwierigkeiten kommen, hier ein Tip: Legen Sie den Stecker einfach neben das Heft, die Anschlüsse für das Kabel nach oben. Stellen Sie nun mit dem Flachkabel die Verbindung zwischen Interface und MZ-80K-Stecker her. Ist Ihnen dies gelungen, können Sie das Interface mit Ihrem MZ-80 verbinden. Dazu ist unten neben dem Netzsteckeranschluß eine hervorstehende Kappe abzuschrauben.
Nachdem Sie nun den Hardwareaufbau geschafft haben, können Sie die Geräte einschalten, und zwar zuerst das Interface und dann den Computer! Verhält sich das System wie vor dem Anschluß der Schnittstelle, so ist alles in Ordnung.
Erscheint auf dem Bildschirm wirres Zeug, und läßt er sich nicht mehr bedienen, so schalten Sie unbedingt das Interface und den MZ-80 aus und überprüfen Sie das Anschlußkabel. Die Masseverbindungen des MZ-80-Bus-Steckers sind sehr wichtig! Sie dienen zur Unterdrükkung von Störungen zwischen den einzelnen Leitungen!

Die Arbeitsweise wird in BASIC durch POKE 64559,XX (64575,XX) festgelegt.
Port A hat die Adresse 64556 (64572)
Port B hat die Adresse 64557 (64573)
Port C hat die Adresse 64558 (64558)

Mit den Befehlen POKE und PEEK können Sie nun mit der Außenwelt in Verbindung treten. Die Ein-/Ausgänge sind TTL-kompatibel, die Ausgänge können auch Darlingtontransistoren ansteuern.

**An der Rückseite des MZ-80K sind Output-Terminals für die Z-80-CPU-Steuer-, Daten- und Adressensignale angebracht. Die Anordnung der Kontaktstifte wird unten gezeigt. Die Terminalzahlen sind so angegeben, wie sie von den Anschlußstellen aus gesehen werden.**

1-Stift-Kennzeichnung




| Befehlswort (XX) | Port A | Port B | Port C | |
|---|---|---|---|---|
| | | | (Bit 0-3) | (Bit 4-7) |
| 128 | A | A | A | A |
| 129 | A | A | E | A |
| 130 | A | E | A | A |
| 131 | A | E | E | A |
| 136 | A | A | A | E |
| 137 | A | A | E | E |
| 138 | A | E | A | E |
| 139 | A | E | E | E |
| 144 | E | A | A | A |
| 145 | E | A | E | A |
| 146 | E | E | A | A |
| 147 | E | E | E | A |
| 152 | E | A | A | E |
| 153 | E | A | E | E |
| 154 | E | E | A | E |
| 155 | E | E | E | E |

A = Ausgang E = Eingang

## Die Software

Die Parallelschnittstelle programmieren Sie entweder über die Adresse 64559 oder 64575. Die nachfolgende Tabelle gibt Ihnen darüber Auskunft:

| | Oberes Terminal A | Unteres Terminal B |
|---|---|---|
| 1 | $D_0$ | GND |
| 2 | $D_1$ | GND |
| 3 | $D_2$ | GND |
| 4 | $D_3$ | GND |
| 5 | $D_4$ | GND |
| 6 | $D_5$ | GND |
| 7 | $D_6$ | GND |
| 8 | $D_7$ | GND |
| 9 | GND | GND |
| 10 | $A_0$ | RESET |
| 11 | $A_1$ | GND |
| 12 | $A_2$ | HALT |
| 13 | $A_3$ | GND |
| 14 | $A_4$ | M1 |
| 15 | $A_5$ | GND |
| 16 | $A_6$ | WR |
| 17 | $A_7$ | GND |
| 18 | $A_6$ | RD |
| 19 | $A_9$ | GND |
| 20 | $A_{10}$ | IORQ |
| 21 | $A_{11}$ | GND |
| 22 | $A_{12}$ | MREQ |
| 23 | $A_{13}$ | GND |
| 24 | $A_{14}$ | INT |
| 25 | $A_{15}$ | GND |

Erich Dohm

# Elektronik für Roboter

Durch meine Arbeit im Volkswagenwerk Wolfsburg als Energieanlagenelektroniker komme ich mit vielen Neuerungen im Bereich der Automatisierung zusammen. Eine besondere Faszination geht von den Industrierobotern oder Hamaten aus, wie sie bei VW genannt werden.

Dies brachte mich schon vor längerer Zeit auf die Idee zu versuchen, etwas Ähnliches nachzubauen. Für mich sollte dieses Gerät allerdings nicht die Funktionen eines vollständigen Hamaten übernehmen, sondern nur Handreichungen, um eventuell Kassetten in einen Recorder einzulegen.

Die mechanischen Bauteile stammen aus dem Bereich des Modellbaues. Der Aufbau erfolgt in mehreren Stufen, so daß das fertige Gerät erst nach längerer Zeit und Versuchen entstand.

Nach einigen Versuchen mit einfachen mechanischen Aufbauten ließ ich mir von einem Bekannten einige mechanische Bauteile für den Anfang herstellen. Gleichzeitig begann ich mit der Entwicklung der DA-Verstärker.

Das Endprodukt ist der aufgezeichnete DA-Verstärker, der auf einer Europakarte aufgebaut und in einem Kartengehäuse untergebracht ist.

Als Endstufe kam eine normale Gegentaktendstufe mit variabler Stromstärke, je nach Größe des Motors, in Betracht. Der Strom wird bestimmt durch die beiden Leistungswiderstände $R_x$. Die Steuerung der Endstufe übernimmt ein OP LM747, in dem zwei Operationsverstärker des Typs LM741 untergebracht sind. Die Umwandlung des digitalen Signales in eine analoge Größe übernimmt der Baustein der LM747 und 74116. Die D-Flipflops mußten sein, da der Verstärker ein Speicherverhalten haben mußte.

Die Wertigkeit der einzelnen Bit ergeben sich aus den den Ausgängen der D-Flipflops nachgeschalteten Widerständen. Über den nachgeschalteten Operationsverstärker wird daraus jeweils eine positive und negative Spannung erzeugt.

Der Abgleich des Verstärkers geschieht, indem man die negative Seite voll ansteuert und den 2,5-kΩ-Einsteller bis auf maximale Ausgangsspannung des Verstärkers einstellt. Danach wird auch die positive Seite angesteuert und der 100-kΩ-Einsteller so eingestellt, bis die Ausgangsspannung Null wird. Der Ausgang des Verstärkers und der Motor ist unbedingt mit einem Kondensator zu entstören, da sonst die D-Flipflops beeinträchtigt werden.

Wenn Motoren mit höheren Strömen angeschlossen sind, ist auf eine ausreichende Kühlung der Endstufentransistoren zu achten. Über den Reseteingang läßt sich der Verstärker unabhängig von der Ansteuerung auf Null stellen.

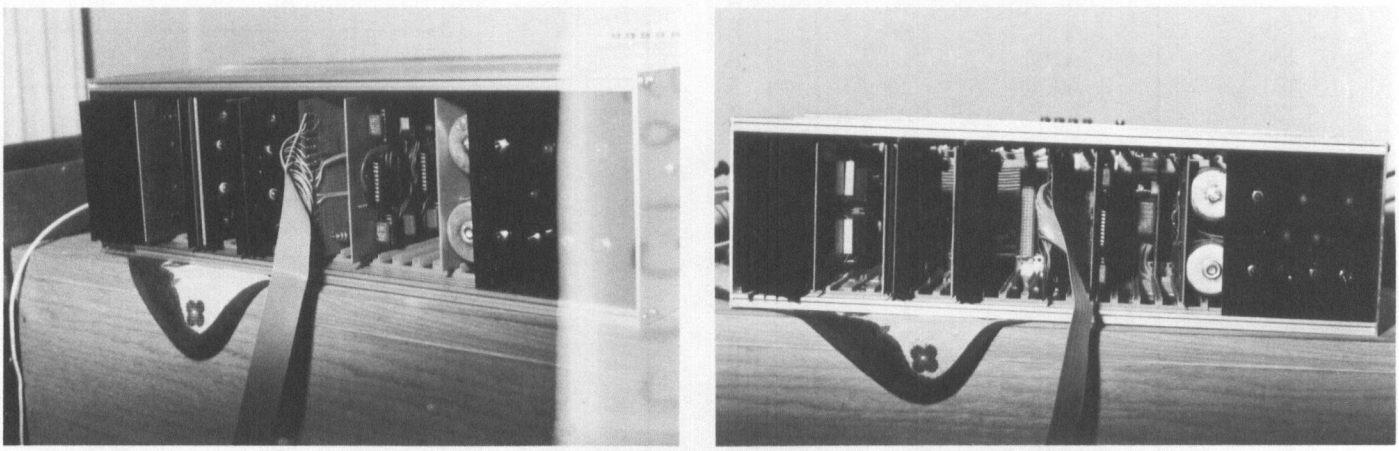
Ein besonderes Problem war, die Stellung der Achsen zu kontrollieren. Dieses ließ sich durch Vor-Rückwärts-Zähler des Typs 74LS193 erreichen. Diese besitzen den Vorteil, daß sie kaskadierbar sind und zwei separate Eingänge besitzen, über die jeweils vor- und zurückgezählt werden kann.

Das Abtasten der Informationen über den Stand der Achsen wird über eine Scheibe, die auf der Motorachse sitzt, bewerkstelligt. Die Scheibe besitzt zwei Einkerbungen und durchläuft zwei Gabellichtschranken. Die auf der Platine der Gabellichtschranke nachgeschalteten Schmitt-Trigger des Typs 74LS132 formen die Signale für die Zähllogik um. Wenn eine Lichtschranke frei wird, ist ein JK-Flipflop über seinen Vorbereitungseingang angesprochen. Nach einem Durchgang gibt das dem JK-Flipflop nachgeschaltete Monoflop einen Impuls einer bestimmten Länge auf die V-R-Zähler und setzt das JK-Flipflop wieder zurück.

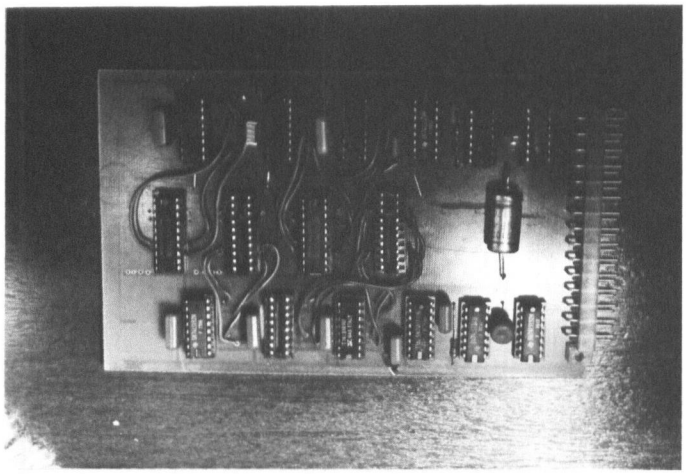
An die Ausgänge der V-R-Zähler sind D-Flipflops mit Tri-State-Ausgängen des Typs 74LS373 angeschlossen, die die Daten der Zähler nach Freigabe auf den Bus des Mikroprozessors freigeben. Das Bit 4 – 7 des Winkelkodierers ist auf zwei D-Flipflops herausgeführt, um eine bessere Kontrolle über den Stand der Achsen zu haben. Im Programm wird zuerst nur der Stand der oberen acht Bit abgefragt. Wenn diese in einem bestimmten Toleranzbereich sind, wird auch die Information der unteren acht Bit abgefragt.

Bei dem Aufbau der Zähler ist darauf zu achten, daß hinter jedem Schaltkreis ein Blockkondensator ist, da sonst die Zähler nicht richtig arbeiten.

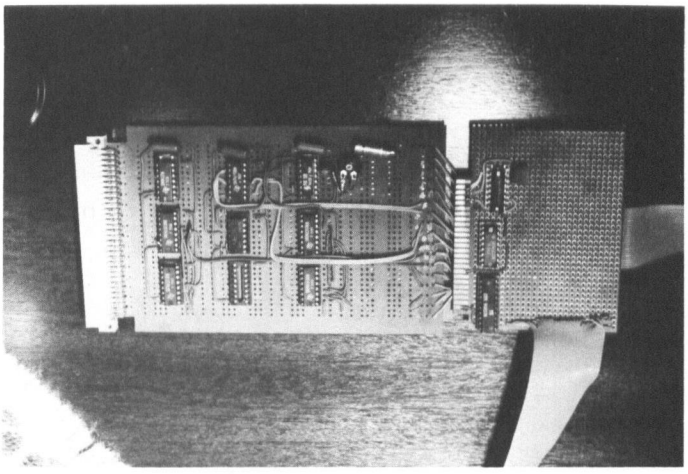
Das Netzteil muß ausreichend dimensioniert sein, da der Anlaufstrom des Motors relativ hoch ist. Außerdem muß eine positive und eine negative Spannung vorhanden sein, die die Gegentaktverstärker versorgt. Ebenso muß für eine ausreichende Entstörung des Netzteiles gesorgt werden, um die Zähler und den Mikroprozessor nicht zu stören. Für die gute Siebung der Gleichspannung sind zwei Kondensatoren mit je 22 000 µF Kapazität erforderlich. Die Ausgangsspannung



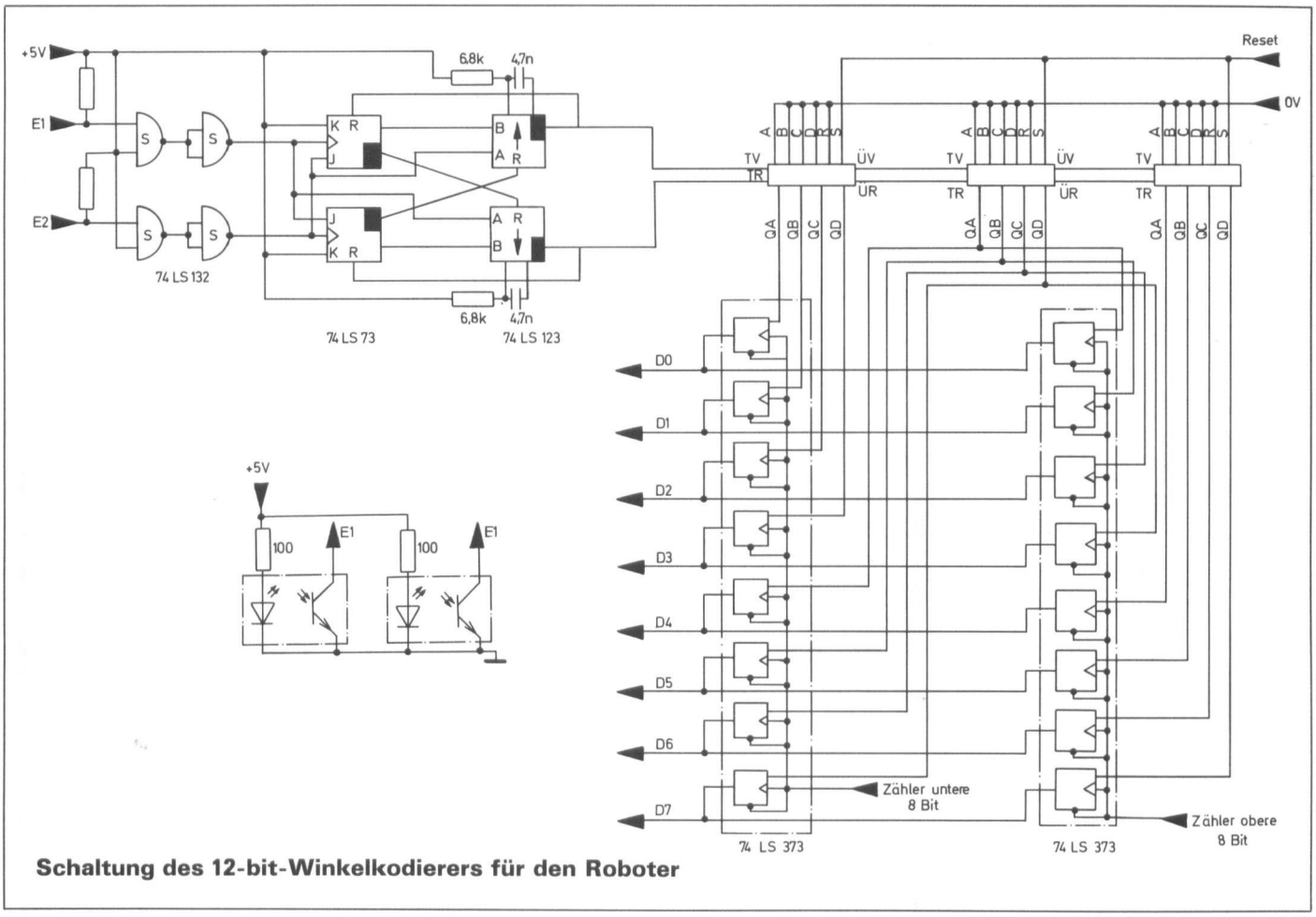
**Gehäuse für die Steckkarten eines selbstgebauten Robotors (Seiten- und Vorderansicht)**



**Platine für den Winkelkodierer mit Zähleinheiten und der Steuerung**



**Interface-Platine für den Winkelkodierer**



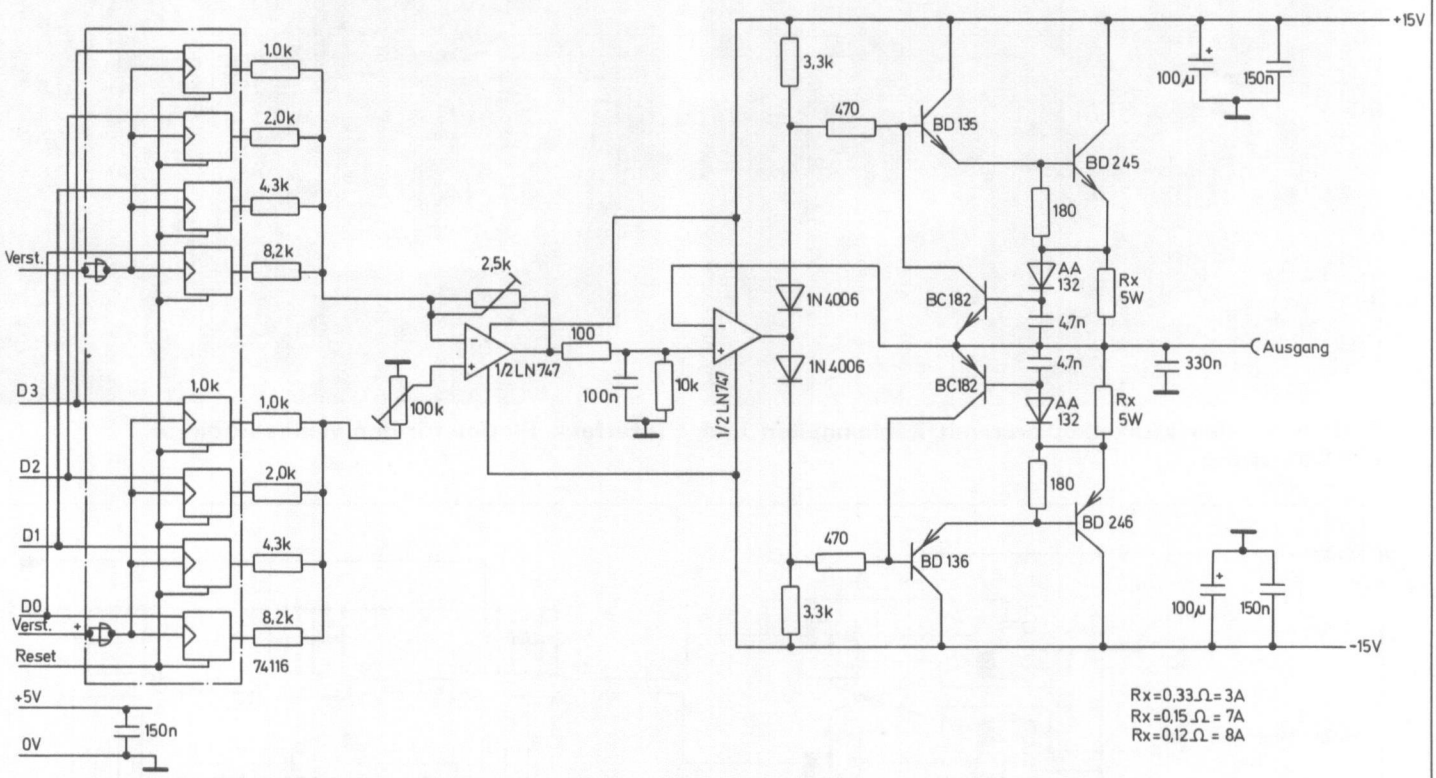

**Schaltung des 12-bit-Winkelkodierers für den Roboter**

wird über zwei Relais geschaltet, um bei einem Fehler die Spannung sofort abschalten zu können (wenn eine Achse in die Begrenzung gefahren ist).

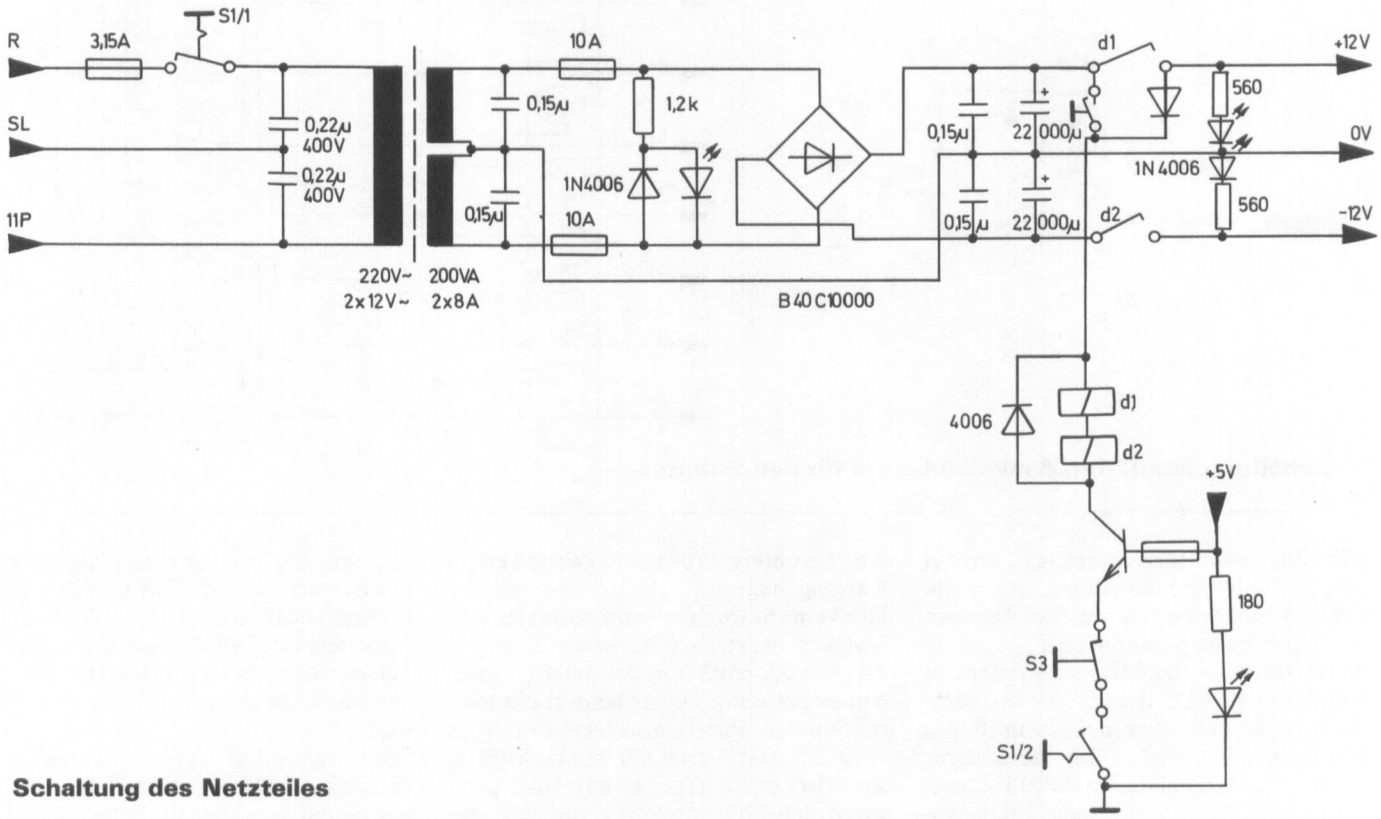
Nach der Wahl des Mikrocomputers, es wurde ein APPLE II Europlus mit 48K-Arbeitsspeicher verwendet, wurde das Interface entwickelt. Ein besonderer Vorteil war hierbei die im APPLE untergebrachten Slots und seinen Taktsignalen. Ein Teil des Interface ist also im APPLE untergebracht, das zweite Teil im Kartengehäuse.

Um keine Störungen beim normalen Ablauf des Prozessors zu erhalten, wurde der Bus des 6502 und der des Kartengehäuses getrennt. Dieses besorgt ein bidirektionaler Bus-Transceiver des Typs 74LS245. Das Signal DS ist ein APPLE-spezifisches Signal, das auf Null geht, wenn der Mikroprozessor auf eine bestimmte Adresse zugreift, die den Slots zugeordnet ist. Über dieses Signal wird der Schaltkreis 74LS245 und der 2-bit-Binärdekoder des Typs 74LS139 angesprochen. Der 2-bit-Binärdekoder gibt einen der vier 3-bit-Binärdekoder frei, die auf dem zweiten Teil des Interface sind.

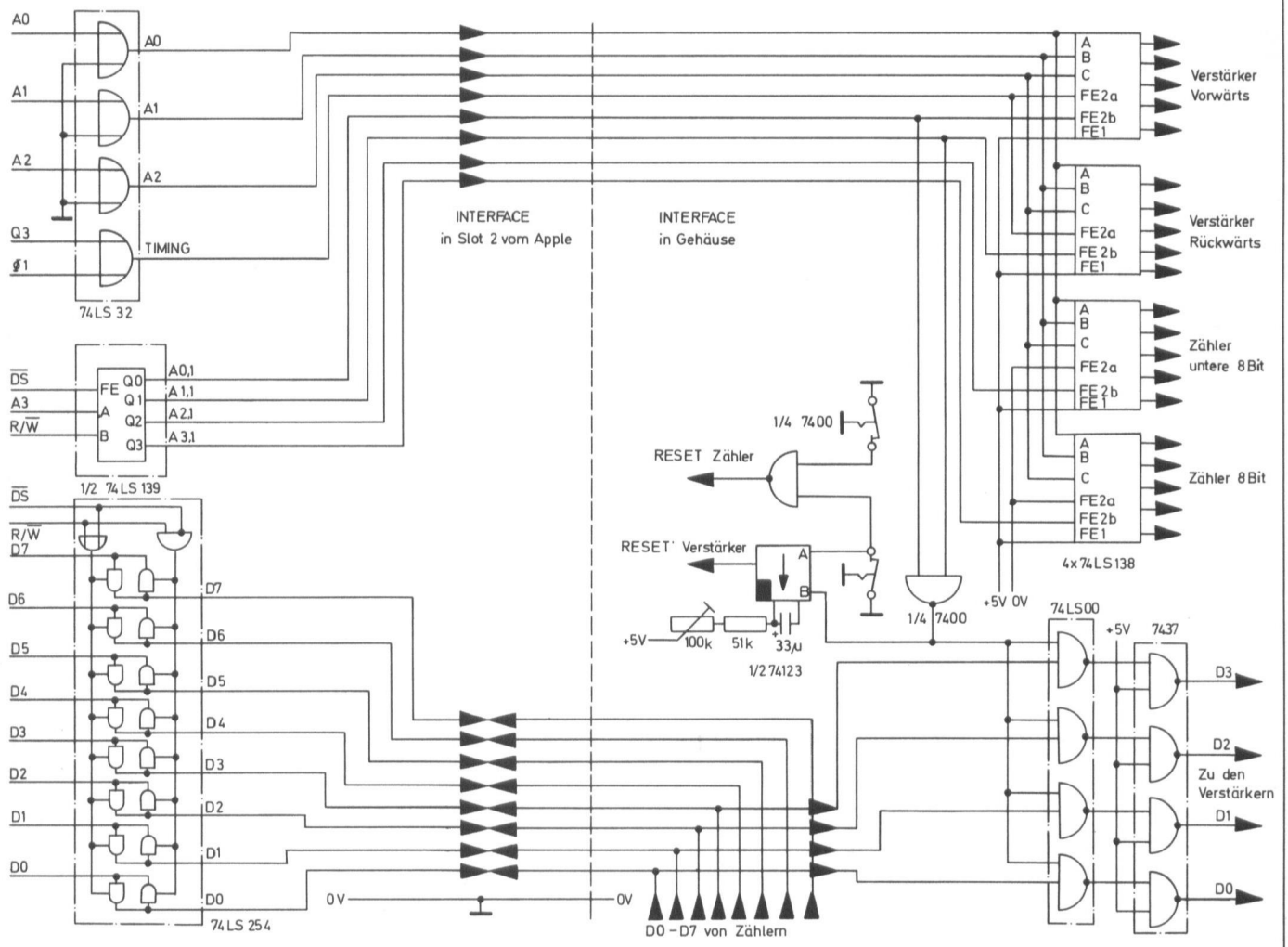
Die unteren drei Adressen werden nun verstärkt und von dem Adressenbus getrennt und dem 3-bit-Binärdekoder zugeführt. Das Timing-Signal wurde gebildet,



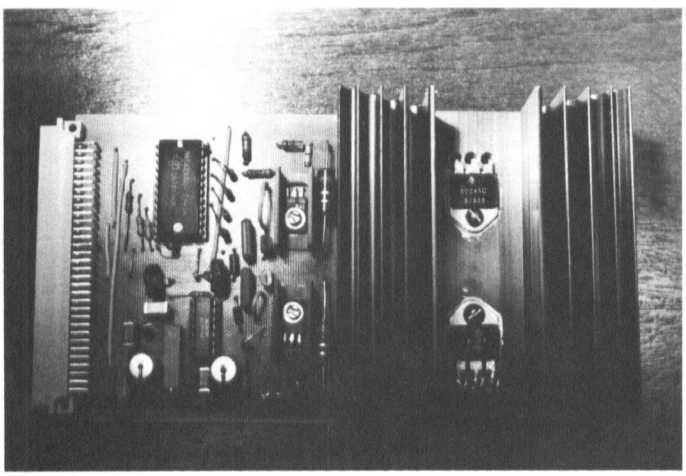

**Schaltung für den Ausgangs-Leistungsteil mit Eingangsspeicher**




**Schaltung des Netzteiles**




**Eingangsschaltung mit Datenspeicherung und Erzeugung der einzelnen Steuerfunktionen**



**Platine für den Ausgangsverstärker mit Kühlblech für die Leistungstransistoren**

da bei einem Takt des 6502 die Daten erst 300 ns vor Ende des Taktes gültig sind, wenn vom 6502 geschrieben wird. Da die D-Flipflops, die auf den DA-Verstärkern sind, schneller sind, würde bei einem Schreibzyklus falsche Daten übertragen. Das D-Flipflop würde verschiedene Zustände während eines Taktes annehmen.

Das Monoflop im Teil 2 des Interface dient dazu, die Verstärker wieder auf Null zu setzen, wenn nach einer bestimmten Zeit kein Schreibsignal mehr gekommen ist, oder wenn das Programm abgebrochen wurde. Ansonsten würden die Motoren unkontrolliert weiterlaufen. Die auf dem Bus des Kartengehäuses nachgeschalteten Bausteine des Typs 74LS00 und 7437 dienen dazu, die Signale für die D-Flipflops auf den Verstärkern umzuformen. Die D-Flipflops sind Standard-TTL-Bausteine.

Die vier 3-bit-Binärdekoder dienen dazu, die Verstärker in positiver oder negativer Richtung anzusprechen und die Daten der Winkelkodierer auszulesen.

Mit dem Reset am Zähler lassen sich die Winkelkodierer auf Null stellen.

Die automatische Regelung der Achsen geschieht über ein Maschinensprache-programm, das von dem BASIC-Hauptprogramm angesprochen wird. Im BASIC-Hauptprogramm werden nur Funktionen wie Konstanten ändern oder lernen, Zusatzinformationen ein- oder ausgegeben und Punkte mit Bezeichnung gespeichert.

Beat Hugi

# EPROM-Programmer für SDK85

Bei den heutigen Preisen für 5-V-EPROM wird ihre Verwendung auch für den Hobbybereich immer interessanter. Das im folgenden vorgestellte Programmiergerät für 2758/2716 oder kompatible EPROM besteht aus einigen wenigen Teilen. Sie umfassen im einzelnen:

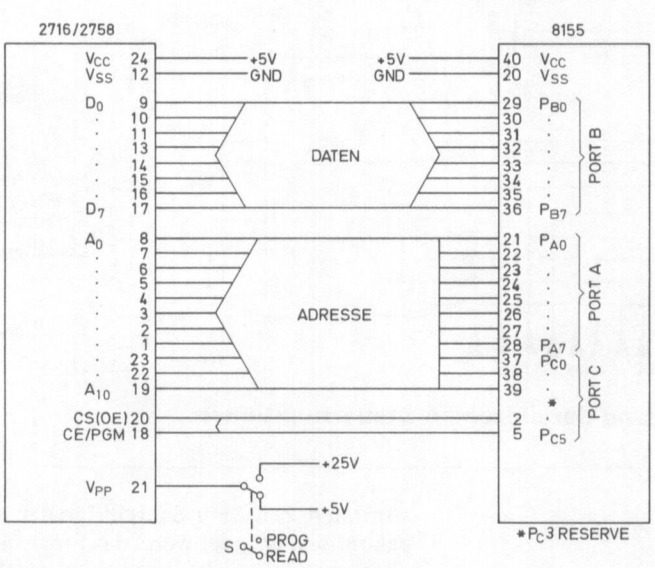
1 IC-Klemmsockel, 24polig, TEXTOOL oder ähnliches,
1 Speisung, 25 V ± 1 V/100 mA,
1 Umschalter 1polig,
1 Gehäuse,
1 Verbindungskabel Programmer – Rechner (PIO).

Die 5-V-Speisung kann direkt vom Rechner bezogen werden.

Die ganze Steuerung des Programmier-/Leseablaufes wird vom Rechner aus softwaremäßig gemacht. Die Originalversion wurde für den Kit SDK85 (8085 CPU) von Intel entwickelt. Sie läßt sich aber durch ein paar Änderungen anhand des Fluß- und des Zeitdiagrammes leicht für jede andere CPU verwenden. Voraussetzung ist jedoch immer ein Parallelkontroller (PIO) ähnlich dem 8155 von Intel. So wurde z.B. für den Einplatinenrechner SBC80/20 von Intel – unter Verwendung seines Monitorprogrammes – ein interaktiver Programmer entwikkelt. Dieser ist nicht nur komfortabel zu bedienen, sondern weist noch den Vorteil auf, daß Programme, die fertig ausgetestet im Speicher sind, direkt in die EPROM kopiert werden können.

Die Software besteht aus zwei Teilen: dem Programmer-(PROG-) und dem Leser-(READ-)Teil.

Da beide Programmteile ähnliche Aufgaben erfüllen, haben sie auch gleiche Eingangsparameter. Der Programmteil



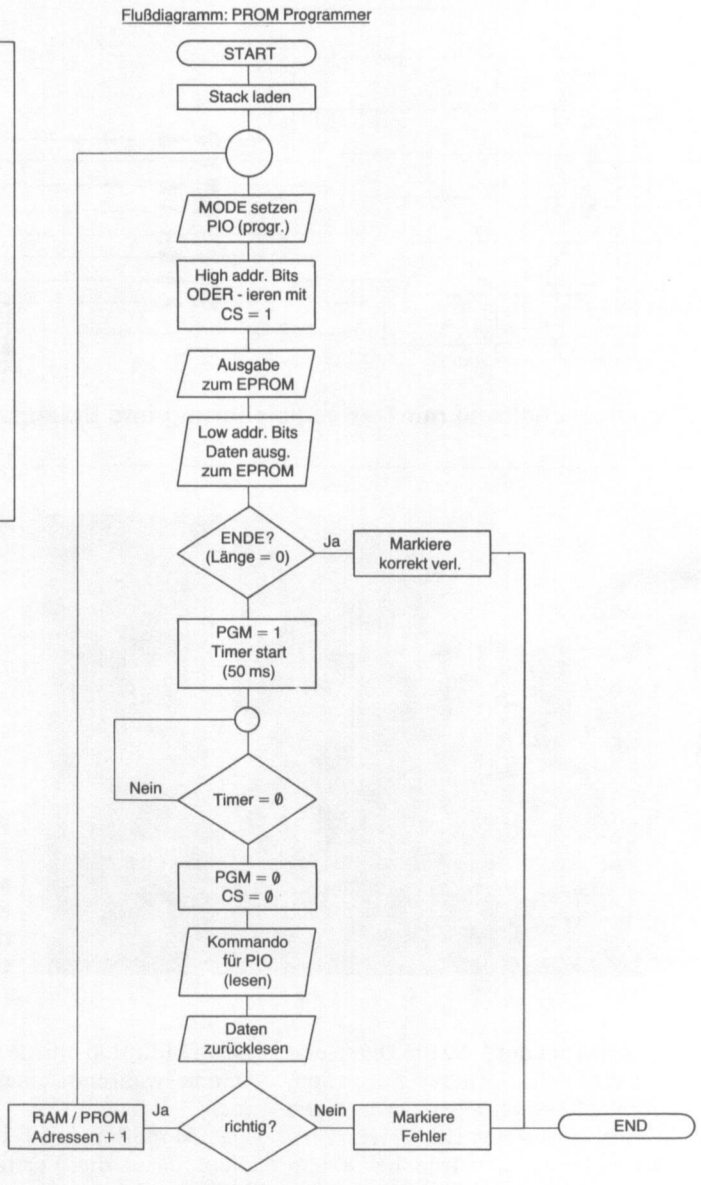

Anschluß EPROM-8155

| 8155 PORT C | $P_{C5}$ | $P_{C4}$ | $P_{C3}$ | $P_{C2}$ | $P_{C1}$ | $P_{C2}$ | Schalter |
|---|---|---|---|---|---|---|---|
| PROM ANSCHL. / MODE | PGM | CS | | A10 | A9 | A8 | $V_{PP}$ |
| READ | 0 | 0 | * | X | X | X | 5V |
| PROGRAM | ⊓ 50ms | 1 | * | X | X | X | 25V |
| VERIFY | 0 | 0 | * | X | X | X | 25V |

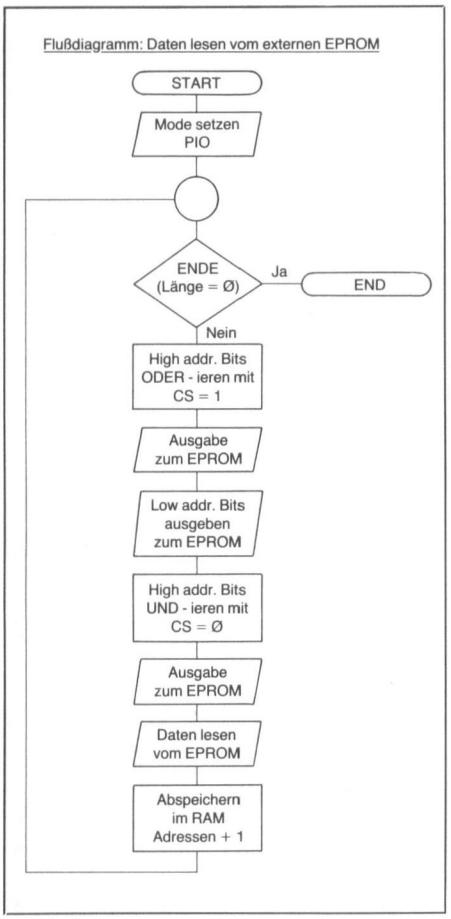
*$P_{C3}$ RESERE

Zusammenhang EPROM, MODES und Ausgänge 8155

**Interface für den EPROM-Programmer und die Bedienungen des 8155-Bausteines**




**Flußdiagramm des EPROM-Programmers für den SDK85**




```
5/11        4.36.72      14-J -80  10:2:19      P   1

                    ; PROM PROGRAMMER FUER 2758/2716 ODER KOMPATIBLE EPROMS
                    ; ***************************************************
                    ;
                    ; FUER CPU 8085 UND PIO 8155/8156 VON INTEL
                    ;
                    ;
                            NLIST
                            LIST
                    ;
                    ; DER INPUT IST FUER PROG UND FUER READ GLEICH.
                    ;
                    ; INPUT: H/L     =       RAM START ADRESSE
                    ;        D/E     =       PROM START ADRESSE
                    ;        B/C     =       LAENGE IN BYTE
                    ;                        FUER  1K EPROM  400H MAX.
                    ;                        FUER  2K EPROM  800H MAX.
                    ;
                    ; DER OUTPUT IST FUER PROG UND READ VERSCHIEDEN.
                    ;
                    ; OUTPUT:        PROG:   Z-FLAG = 1  KEIN PROGRAMMIERUNGS FEHLER
                    ;                         Z-FLAG = 0  PROGRAMMIERUNGS FEHLER
                    ;                READ:   -----
                    ;
                    ;       GLEICHSETZUNGEN
                    ;       ***************
                    ;
CC0F                PMODE   EQU     0FH     ; IOE1 PORT A/B/C = AUSGABE
CC0D                RMODE   EQU     0DH     ; IOE1 PORT A/C   = AUSGABE
                                                PORT B      = EINGABE
CC10                CS1     EQU     10H     ; CS = 1
CC07                CS0     EQU     07H     ; CS = 0
CC20                CSR     EQU     20H     ; KOMMANDO UND STATUS REG IOE1
CC21                PA      EQU     21H     ; IOE1 PORT A
CC22                PB      EQU     22H     ; IOE1 PORT B
CC23                PC      EQU     23H     ; IOE1 PORT C
CC20                PGM1    EQU     20H     ; PGM = 1
CC1F                PGM0    EQU     1FH     ; PGM = 0
2626                PTIME   EQU     9766    ; PGM = 1 ZEIT ( 50MS )
20C2                TOS     EQU     20C2H   ; STACK ANFANG
                    ;
                    ;
                            NAME    PROG85
                            INTERN  PROG
                            INTERN  READ
                            RSECT   DCODE
                    ;
                    ;
                    ;       PROGRAMM TEIL
                    ;       *************
                    ;
CC00"               PROG:                   ; EINSPRUNGADRESSE FUER PROG
CC00" 31 C2 20              LXI     SP,TOS  ; LADE STACK POINTER
```

**Programm für die EPROM**

PROG kopiert einen Speicherbereich (Startadresse und Länge) in ein EPROM (Startadresse). Der Programmteil READ kopiert einen EPROM-Bereich (Startadresse und Länge) in den Speicher des Rechners (Startadresse).

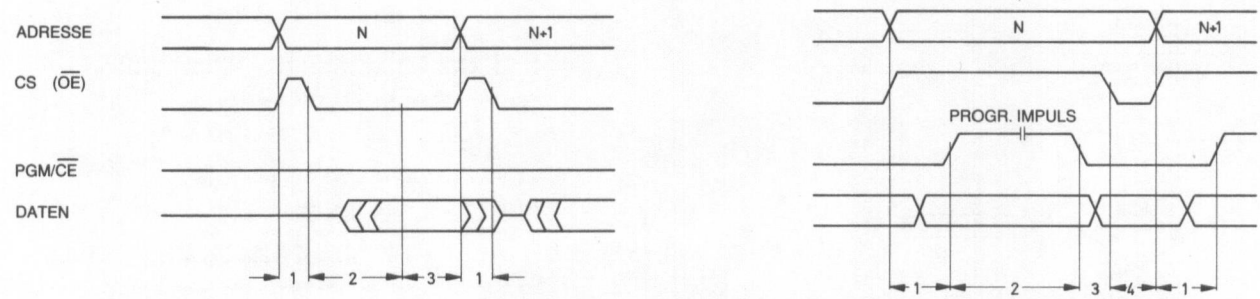
Der zeitliche Ablauf für den Programmier- und Lesevorgang ist den Zeitdiagrammen zu entnehmen. Aus der Abbildung ist ersichtlich, wie das EPROM an den PIO8155 des Rechners angeschlossen ist. Port B dient als Datenport, sowohl für Eingabe (lesen, prüfen) als auch für Ausgabe (programmieren). Die Adressen werden über Port A und Port C (PC0 bis PC2) eingestellt. PC4 steuert $\overline{CS}$ und PC5 PGM des EPROM. PC3 ist nicht verwendet, könnte aber für eine der folgenden Aufgaben dienen:

- automatische Umschaltung PROG/READ, d.h. 25 V/5 V auf Anschluß 21 des EPROM. Dazu wäre allerdings einige zusätzliche Elektronik notwendig. Das Umschalten wird hier über den Schalter SW von Hand gemacht.
- Als zusätzliches Adressenbit für 4K-EPROM.

Eine nähere Beschreibung des Programmes erübrigt sich, da das Programm selbst genügend gut dokumentiert ist. Als

```
5/11        4.36.72      14-J -80  10:2:21      P   1-1

CC03" 3E 10                 MVI     A,CS1   ; CS AUF LOG 1
CC05" B2                    ORA     D       ; LOG. ODER MIT HIGH ADR. BIT'S
CC06" 57                    MOV     D,A     ; SAVE HIGH ADR.+ CS1 = 1
CC07"               PROGL:                  ; PROGRAMMIERUNGS SCHLEIFE
CC07" 3E 0F                 MVI     A,PMODE ; KOMMANDO FUER IOE1
CC09" D3 20                 OUT     CSR     ; AUSGABE NACH CSR IOE1
CC0B" 7A                    MOV     A,D     ; HOLE HIGH ADR.+ CS1 = 1
CC0C" D3 23                 OUT     PC      ; AUSGABE ZUM EPROM
CC0E" 7B                    MOV     A,E     ; HOLE LOW ADR. BYTE
CC0F" D3 21                 OUT     PA      ; AUSGABE ZUM EPROM
CC11" 7E                    MOV     A,M     ; DATEN VOM RAM
CC12" D3 22                 OUT     PB      ; AUSGABE ZUM EPROM
CC14" 0B                    DCX     B       ; LAENGE = 0 ?
                            JX5     ENDPV   ; JA, ENDE
CC15" FD                    DB      0FDH    ; OP-CODE
CC16" 3F 00"                DW      ENDPV   ; ZIEL
CC18" 3E 20                 MVI     A,PGM1  ; PGM AUF LOG 1
CC1A" B2                    ORA     D       ; LOG. ODER MIT HIGH ADR. BIT'S
CC1B" D3 23                 OUT     PC      ; AUSGABE ZUM EPROM
CC1D" D5                    PUSH    D       ; SAVE HIGH PROM ADR
CC1E" 11 26 26              LXI     D,PTIME ; LADE PROGR ZEIT
                    ;
CC21"               TLOOP:                  ; PROGR ZEIT SCHLAUFE
CC21" 1B                    DCX     D       ; ENDE ?
                            JNX5    TLOOP   ; NO=>
CC22" DD                    DB      0DDH    ; OP-CODE
CC23" 21 00"                DW      TLOOP   ; ZIEL
                    ;
CC25" D1                    POP     D       ; RESTORE PROM ADR
CC26" 3E 1F                 MVI     A,PGM0  ; PGM AUF LOG 0
CC28" A2                    ANA     D       ; LOG. UND MIT LOW ADR. BYTE
CC29" D3 23                 OUT     PC      ; AUSGABE ZUM EPROM
CC2B" 3E C7                 MVI     A,CS0   ; CS AUF LOG 0
CC2D" A2                    ANA     D       ; LOG. UND MIT LOW ADR. BYTE
CC2E" D3 23                 OUT     PC      ; AUSGABE ZUM EPROM
CC30" 3E 0D                 MVI     A,RMODE ; KOMMANDO FUER IOE1
CC32" D3 20                 OUT     CSR     ; AUSGABE NACH CSR IOE1
CC34" DB 22                 IN      PB      ; LIES PROGR DATA BYTE
CC36" BE                    CMP     M       ; VERGLEICH MIT RAM
CC37" C2 43 00"             JNZ     FEHLP   ; ABBRUCH PROGR VORGANG
CC3A" 13                    INX     D       ; PROM ADR
CC3B" 23                    INX     H       ; RAM ADR
CC3C" C3 07 00"             JMP     PROGL   ; PROGRAMMIER SCHLAUFE
                    ;
CC3F"               ENDPV:                  ; PROGRAMM EXIT, KEIN FEHLER
CC3F" 3E 01                 MVI     A,1     ; MARKIERE KORREKTEN PROGR VERLAUF
CC41" B7                    ORA     A       ;
CC42" C9                    RET             ;
                    ;
CC43"               FEHLP:                  ; PROGRAMM EXIT, FEHLER IM PROM
CC43" AF                    XRA     A       ; MARKIERE FEHLERHAFTEN PROGR VERLAUF
CC44" C9                    RET             ;
```

**Programmablauf zwischen CC03 und CC44**




ZEITABSCHNITT: 1 ADRESSE AUSGEBEN, $\overline{CS}$ = 1
2 $\overline{CS}$ = O (DELAY BIS DATEN STABIL)
3 DATEN LESEN UND ADRESSE + 1

ZEIT DIAGRAMM LESEN

ZEITABSCHNITT: 1 ADRESSE AUSGEBEN, $\overline{CS}$ = 1
2 PROGRAMMIERER IMPULS 50 ms
3 PIO UMSTEUERN ZUM LESEN, $\overline{CS}$ = 0
4 DATEN LESEN UND PRÜFEN

ZEIT DIAGRAMM PROGRAMMIEREN

**Lesen der Daten von einem externen EPROM**

```
5/11          4.36.72        14-J  -80  10:2:21       P     1-2

                    ;
                    ;
                    ;
                    ;       LIES DATEN VOM EXTERNEN EPROM
                    ;       ******************************
                    ;
C045"               READ:                       ; EINSPRUNGADRESSE FUER READ
C045" 3E 0D                 MVI     A,RMODE ; KOMMANDO FUER IO£1
C047" D3 20                 OUT     CSR     ; AUSGABE NACH CSR IO£1
                    ;
C049"               READL:                      ; READ LOOP
C049" 0B                    DCX     B       ; LAENGE = 0 ?
                            JX5     READE   ; YES ->
C04A" FD                    DB      0FDH    ; OP-CODE
C04B" 62 00"                DW      READE   ; ZIEL
C04D" 3E 10                 MVI     A,CS1   ; CS AUF LOG 1
C04F" B2                    ORA     D       ; LOG. ODER MIT HIGH ADR. BIT'S
C050" D3 23                 OUT     PC      ; AUSGABE ZUM EPROM
C052" 7B                    MOV     A,E     ; HOLE LOW ADR. BYTE
C053" D3 21                 OUT     PA      ; AUSGABE ZUM EPROM
C055" 3E 07                 MVI     A,CS0   ; CS AUF LOG 0
C057" A2                    ANA     D       ; LOG. UND MIT HIGH ADR. BYTE
C058" D3 23                 OUT     PC      ; AUSGABE ZUM EPROM
C05A" DB 22                 IN      PB      ; DATEN LESEN VOM EPROM
C05C" 77                    MOV     M,A     ; ABSPEICHERN IM RAM
C05D" 23                    INX     H       ; RAM ADR
C05E" 13                    INX     D       ; PROM ADR
C05F" C3 49 00"             JMP     READL   ;  LIES NAECHSTES BYTE
C062"               READE:                      ; PROGRAMM EXIT
C062" C9                    RET             ;
                            END
```

weitere Hilfe dienen die Flußdiagramme.

Folgende Anmerkungen sind jedoch zu machen:

- Der 50-ms-Impuls wird durch Dekrementieren eines Doppelregisters programmiert und ist somit von der Taktfrequenz des verwendeten Rechners abhängig. Er muß also nötigenfalls angepaßt werden (Toleranz: ±5 ms), d.h., PTIME muß verändert werden.
- Jedes Byte wird unmittelbar nach dem Programmiervorgang zurückgelassen und mit dem Speicherinhalt verglichen. Im Fehlerfall wird der Programmiervorgang abgebrochen. Ein Prüfen nach dem Programmieren erübrigt sich damit.
- Mit den beiden Befehlen JX5 und JNX5 kann nach einem Doppelregister-Dekrement/-Inkrement ein Wechsel von 0000H nach FFFFH oder von FFFFH nach 0000H festgestellt werden. Es sind jedoch Befehle, die beim 8085 nur inoffiziell und beim 8080 überhaupt nicht vorhanden sind. Sie können jedoch durch die folgenden Makros leicht nachgebildet werden:

```
JX5:    MACRO   ADR        JNX5:   MACRO   ADR
        PUSH    H                  PUSH    H
        LXI     H,1                LXI     H,1
        DAD     B                  DAD     B
        POP     H                  POP     H
        JC      ADR                JNC     ADR
        ENDM                       ENDM
```

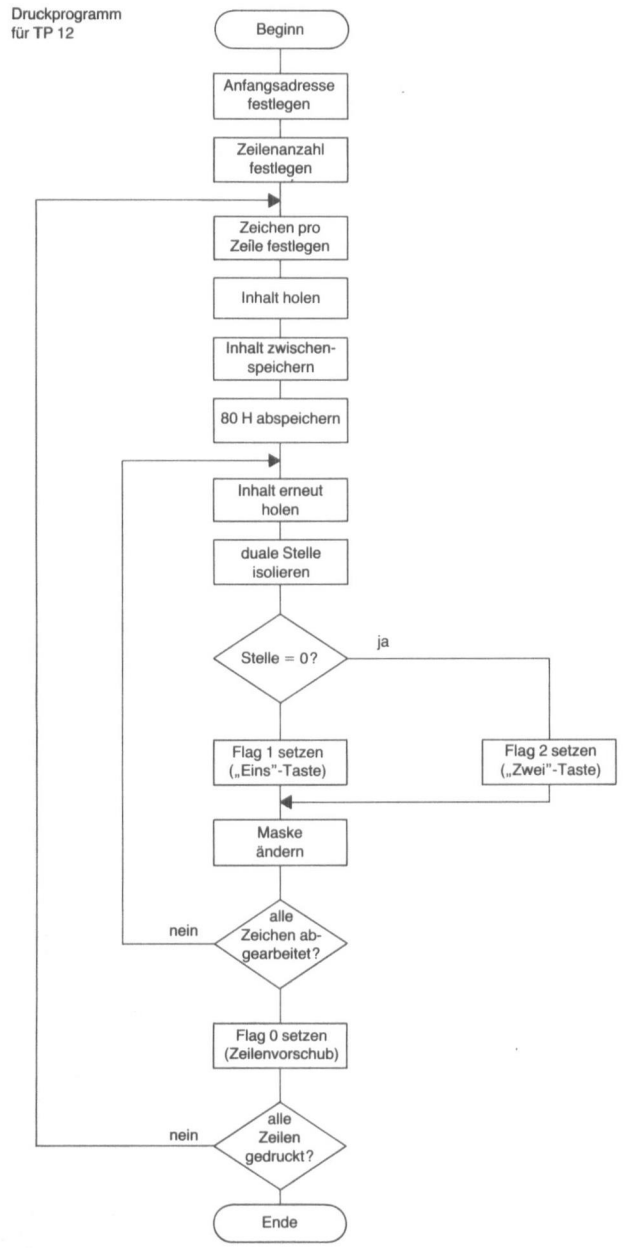
- $\overline{CS}$ entspricht der neueren Bezeichnung $\overline{OE}$.

Eduard Stiebel

# Speicher auslesen mit druckendem Kaufhausrechner

## Der druckende Rechner TP 12

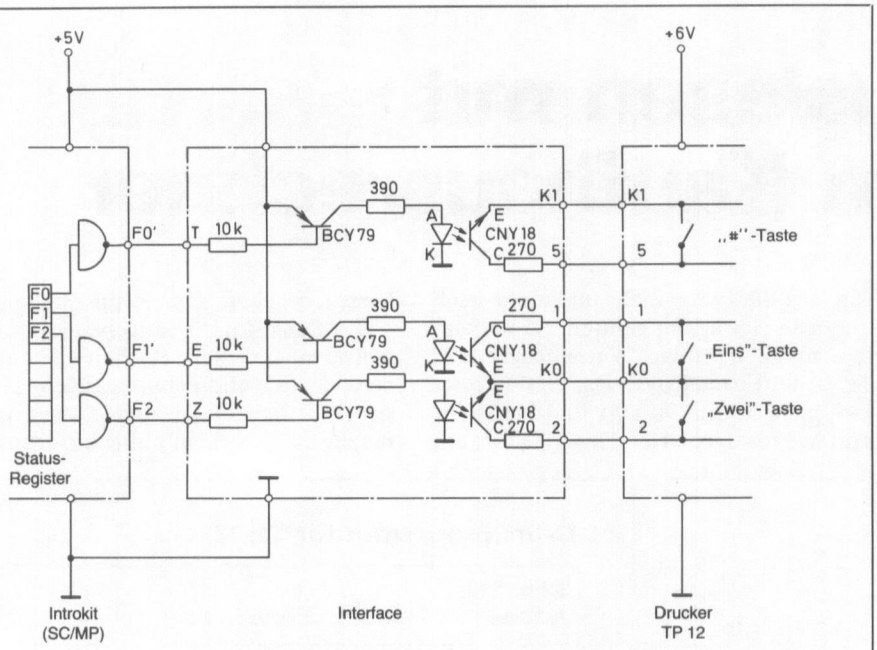
Seit einiger Zeit bieten auch Kaufhäuser druckende Rechner in verschiedenen Größenordnungen an. Dabei kommt dem Hobbycomputerfreund sehr entgegen, daß die Preise mehr und mehr nach unten in Bewegung geraten. So verkaufen zur Zeit einige Warenhäuser den TP 12 der Firma Royal. Das Gerät kostet zwischen 50 und 90 DM und bietet für seinen Preis recht viel. Der TP 12 ist ein Thermodrucker, der zwölf Stellen in 5 × 7-Punkt-Matrix ausgehen kann. Neben den üblichen Rechenoperationen der vier Grundrechnungsarten lassen sich mit dem Gerät auch Zahlen ohne rechnerische Verknüpfung ausdrucken.




**Flußdiagramm für den druckenden Rechner**

### Druckprogramm für TP 12

| Listing Adresse | | Inhalt | | Bemerkungen |
|---|---|---|---|---|
| OF40 | 41 | C4 | OF | Anfangsadresse (H) |
| 42 | | 35 | | |
| 43 | 44 | C4 | 90 | Anfangsadresse (L) |
| 45 | | 31 | | |
| 46 | 47 | C4 | OC | Zeilen (12) |
| 48 | 49 | C8 | 45 | |
| 4A | 4B | C4 | 08 | Zeichen pro Zeile (8) |
| 4C | 4D | C8 | 40 | |
| 4E | 4F | C5 | 01 | |
| 50 | 51 | C8 | 3B | |
| 52 | 53 | C4 | 80 | |
| 54 | 55 | C8 | 36 | |
| 56 | 57 | CO | 35 | |
| 58 | 59 | DO | 32 | |
| 5A | 5B | 98 | 04 | |
| 5C | 5D | C4 | 02 | |
| 5E | 5F | 90 | 02 | |
| 60 | 61 | C4 | 04 | |
| 62 | | 07 | | |
| 63 | 64 | 8F | 2F | |
| 65 | 66 | CO | 25 | |
| 67 | | 1E | | |
| 68 | 69 | C8 | 22 | |
| 6A | 6B | C4 | 00 | |
| 6C | | 07 | | |
| 6D | 6E | 8F | 2F | |
| 6F | 70 | B8 | 1D | |
| 71 | 72 | 9C | E3 | |
| 73 | 74 | C4 | 01 | |
| 75 | | 07 | | |
| 76 | 77 | 8F | FF | |
| 78 | 79 | C4 | 08 | |
| 7A | 7B | C8 | 0F | |
| 7C | 7D | C4 | 00 | |
| 7E | | 07 | | |
| 7F | 80 | 8F | FF | |
| 81 | 82 | B8 | 08 | |
| 83 | 84 | 9C | FA | |
| 85 | 86 | B8 | 08 | |
| 87 | 88 | 9C | C1 | |
| 89 | | 3F | | |
| 8A | | Z (Zeit) | | |
| 8B | | S (Selektion) | | |
| 8C | | W (WERT) | | |
| 8D | | C (Zeichen) | | |
| 8E | | L (Zeilen) | | |




**Interface-Schaltung zwischen Introkit und Drucker**

```
          0 0
22 222 221 # (1)
22 222 212 # (2)
22 222 211 # (3)
22 222 122 # (4)

        0 0
22 222 221 # (Kode)
22 222 211 #
22 222 111 #
22 221 111 #
22 211 111 #
22 111 111 #
21 111 111 #
11 111 111 #

          0 0
  C   4
11 222 122 #
  0   2
22 222 212 #
  0   7
22 222 111 #
  8   F
12 221 111 #
  F   F
11 111 111 #
  C   4
11 222 122 #
  0   0
22 222 222 #
  0   7
22 222 111 #
  8   F
12 221 111 #
  F   F
11 111 111 #
  9   0
12 212 222 #
  F   4
11 112 122 #
```

**Tabelle für die Ansteuerung**

## Anschluß an einen Mikrocomputer (SC/MP)

Diese Eigenschaften des TP 12 haben den Autor veranlaßt, zu versuchen, das Gerät an ein Introkit (SC/MP) anzuschließen. Da das Gehäuse des TP 12 sehr leicht zu öffnen ist und im Innenraum noch genügend Platz ist, lassen sich leicht ein paar Drähte oder ein dünnes Kabel nachziehen. Über die Öffnungen im Batteriefach kann das sogar ohne Bohrarbeit geschehen. Ein paar andere Schönheitsfehler stellen sich jedoch ein. Die unmittelbare Darstellung hexadezimaler Zeichen ist schwer möglich; vorlaufende Nullen werden vom Gerät unterdrückt; die Darstellung der Zahlen in Dreiergruppen stört. Ein paar Kompromisse sind also notwendig.

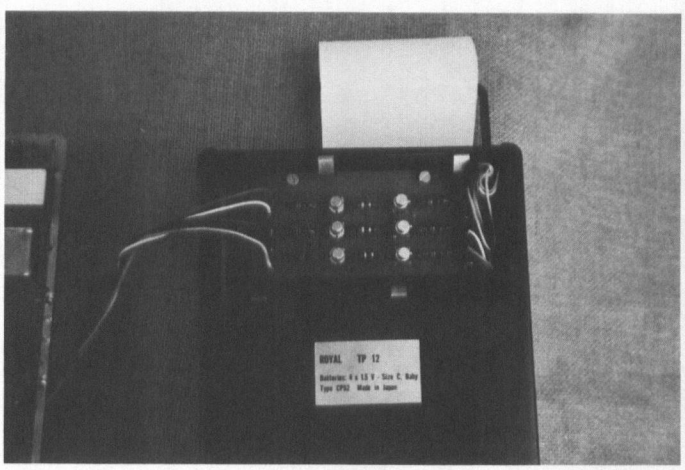
Die Dreiergruppendarstellung der Zahlen kann man ignorieren. An die Stelle der Dualzahlen 0 und 1 können die Ziffern 2 und 1 treten, so daß der Drucker den Speicherinhalt trotzdem – in einer leicht modifizierten Form – dual ausgeben kann.

Der Anschluß dreier Flags genügt, um dies zu erreichen. Flag 1 und Flag 2 steuern die „Eins"- bzw. „Zwei"-Taste an, und Flag 0 löst über die „#"-Taste den Zeilenvorschub aus.

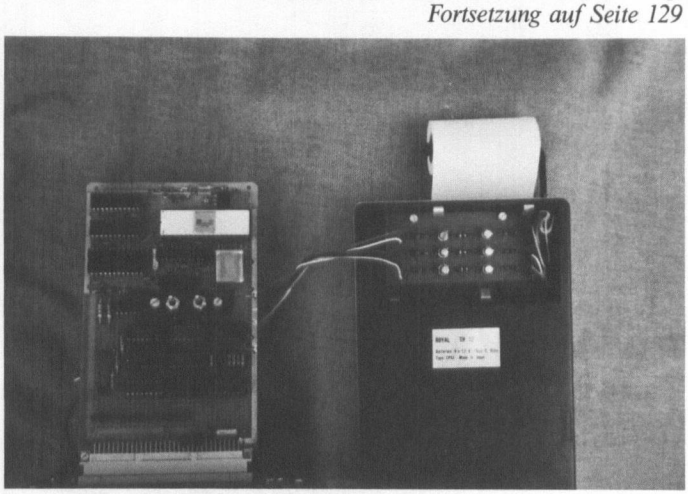
## Das Interface

Ein kleines Interface verbindet den Mikrocomputer mit der Tastatur des TP 12. Die Tastatur aufzubrechen lohnt sich fast nicht, da die Anschlußpunkte gut lesbar bezeichnet sind. So liegen an 1 und KO die „Eins"-Taste, an 2 und KO die „Zwei"-Taste und an 5 und K1 die „#"-Taste. Zum Überprüfen der Funktionen können drei weitere Hilfstasten auf der Interface-Karte für den Anfang recht hilfreich sein. Für die eigentliche Ankopplung an den Mikrocomputer eig-

*Fortsetzung auf Seite 129*



**Geöffnetes Druckergehäuse mit Interface**



**Introkit mit dem drückenden Rechner**

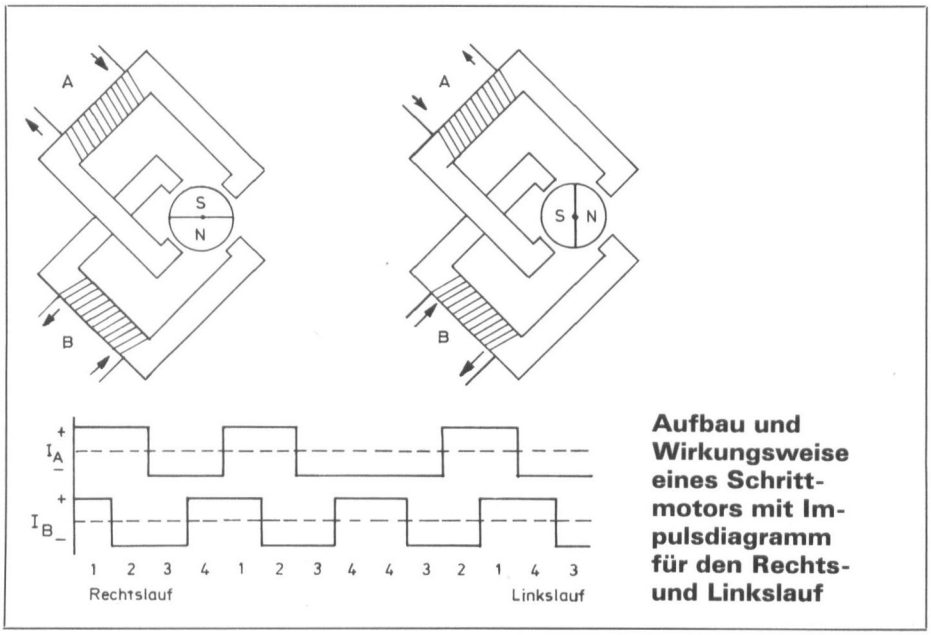
Peter Bentz

# Ein mikrocomputergesteuerter Selbstbauplotter

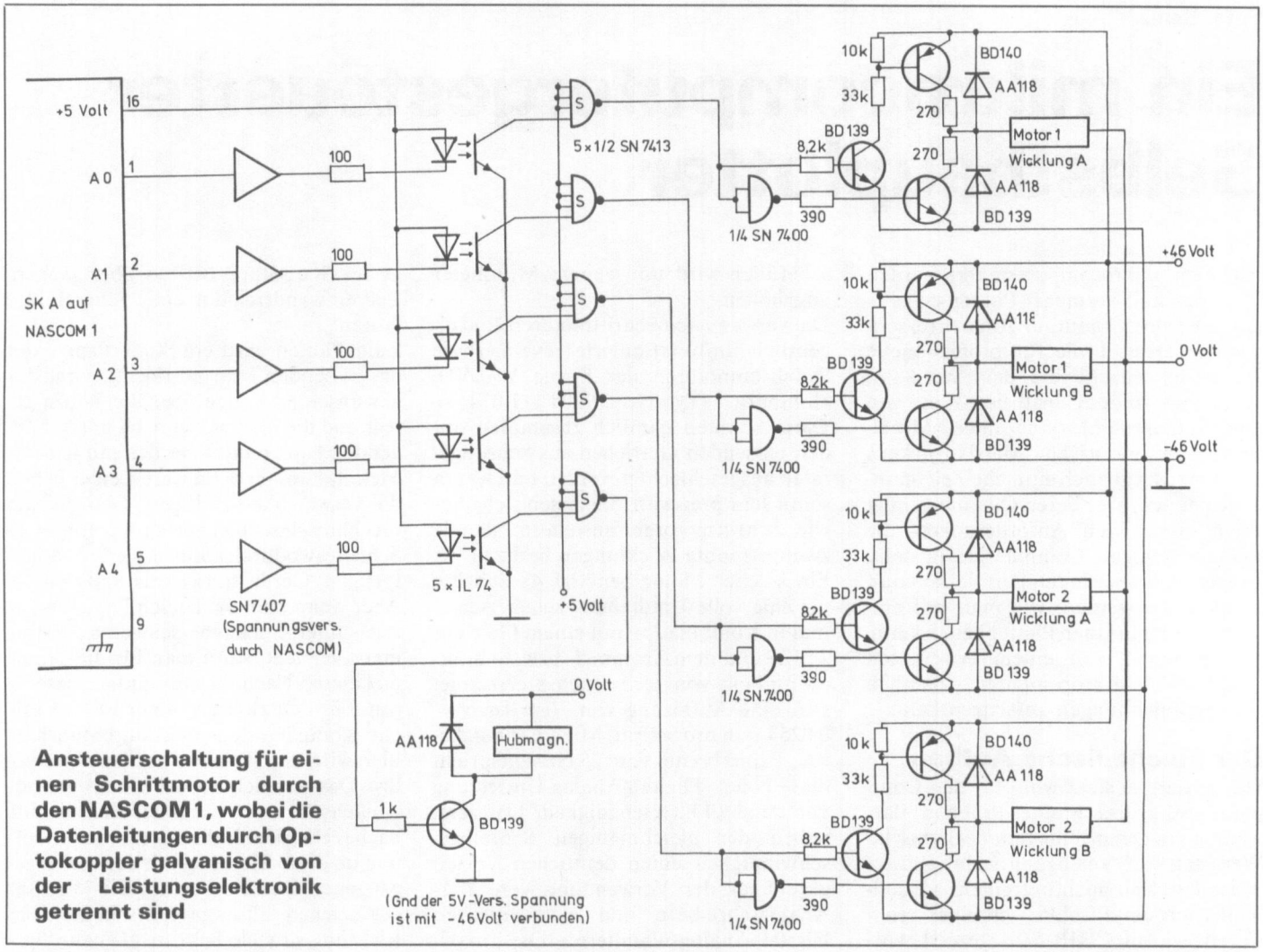
Auf dem Mikrocomputermarkt werden in letzter Zeit vermehrt Plotter angeboten, die jedoch kaum für Amateure, sondern in erster Linie für professionelle Anwender gedacht sind; denn wer kann schon nur so zum Spaß den Preis von einigen tausend Mark bezahlen? Mit einigen preisgünstigen Matrixdruckern kann man zwar auch einfache Zeichnungen erstellen, aber deren Qualität genügt nicht immer den Anforderungen, die man an eine gute Computergrafik stellt. In diesem vorliegenden Beispiel soll nun beschrieben werden, wie man mit einfachen Mitteln einen Plotter bauen kann, der, gesteuert von einem erweiterten NASCOM-1-Mikrocomputer, ziemlich präzise Zeichnungen anfertigen kann.

## Der mechanische Aufbau

Dieses Kapitel stellt wohl für den Computerbastler das größte Problem dar, wenn nicht gerade eine feinmechanische Werkstatt zur Verfügung steht. Ersatzweise läßt sich auch mit einem Modellbaukasten ein Flachbettschreiber realisieren, wie in CHIP 5/81 gezeigt wurde, aber hier soll als weitere Alternative ein Trommelplotter vorgestellt werden.
Aus einem ausrangierten Linienschreiber „Speedomax Typ G" der Firma Leeds & Northrup Co., Philadelphia, USA, wurde die überflüssig gewordene Mechanik und die ohnehin defekte Elektronik ausgebaut. Für den Papiertransport wurde als einfachste Lösung ein Schrittmotor mit Getriebe direkt auf der Trommelachse befestigt. Für die Bewegung des Zeichenstiftes sorgt ein zweiter bauartgleicher Motor mit Getriebe. Ein Schnurrad, das zweckmäßigerweise den gleichen Durchmesser wie die Trommel hat (48,5 mm), sitzt auf der Getriebeachse und zieht mit einem dünnen Stahlseil (Violinsaite) über vier Umlenkrollen einen Schlitten. Dieser Schlitten wird durch einen 7 mm starken Rundstab parallel zur Trommelachse geführt. Er gleitet dabei über eine zweite Laufschiene mit Rechteckquerschnitt. Die Laufschiene kann von einem Elektrohubmagneten über ein Gestänge etwas um die Längsachse gedreht werden, so daß sich die eine Auflagestelle des Schlittens hebt. Dadurch wird der Schlitten nach oben gekippt und der darin eingeschraubte Tuschefüller wird um wenige Millimeter angehoben.
Die verwendeten Schrittmotoren sind eigentlich umfunktionierte reversierbare Synchronmotoren der Firma VALVO, Hamburg (Typ-Nr. 9904 111 07413). Diese konnten nämlich zusammen mit den passenden Getrieben aus Schrottgeräten ausgeschlachtet werden und waren somit sehr preiswert. Sie lassen sich aber wie Schrittmotoren ansteuern, da sie zwei getrennte Wicklungen besitzen.
Ein solcher Motor benötigt 48 Schritte für eine volle Umdrehung seiner Achse, und in Kombination mit einem Getriebe 1 : 125 und dem Trommel- bzw. Schnurradumfang von sechs Inches errechnet sich eine Auflösung von $^1/_{1000}$ Inch = 0,0254 mm pro Schritt. Mit einer nutzbaren Papierbreite von 25 cm entspricht dies einer Digital/Analog-Umsetzung von rund 14 bit Genauigkeit. Das zeigt neben der gleichmäßigen Schreibgeschwindigkeit einen deutlichen Vorteil gegenüber der Verwendung von *X/Y*-Analogschreibern und elektronischen Digital/Analog-Wandlern. Die maximale Schrittfrequenz der Motoren beträgt ca. 280 Hz, was je nach Schreibrichtung einer Geschwindigkeit von 5,2 bis 7,2 mm pro Sekunde entspricht. Dabei ist schon eine gewisse Sicherheitsspanne berücksichtigt, denn die Motoren müssen ja im Start-Stopp-Betrieb über längere Zeit einwandfrei, d.h. ohne Schrittfehler laufen.
Zum Plotten wird ein Stück Papier mit der passenden Transportlochung und der gewünschten Länge über die Walze gelegt und die beiden frei nach unten hängenden Papierenden werden mit je einer Metallklammer beschwert. Leider haben die verwendeten „billigen" Getriebe ein mechanisches Spiel von ca. 0,5 mm = 18 Schritte, wodurch anfangs die Freude am fertigen Gerät stark gedämpft wurde. Aber wenn man die beiden Papierenden mit unterschiedlich schweren Klammern versieht, kann man für die Trommel diesen Nachteil ganz einfach beseitigen. Für den zweiten Motor ist eine solche Kompensation nicht durchführbar; hier wird das Problem per Software gelöst. Das geschieht so, daß ein Programm zunächst überprüft, ob der Zeichenstift nach rechts oder links bewegt werden soll und ob diese Richtung mit der vorhergegangenen übereinstimmt. Bei Wechsel der Zeichenrichtung werden in die neue Richtung so viele Schritte (18) durchgeführt, daß sich der Zeichenstift gerade noch nicht bewegt. Nun wird erst der eigentliche Zeichenvorgang ausgeführt und damit wird auch der Stift sofort mitbewegt, als ob er überhaupt keinen Spielraum hätte.




**Aufbau und Wirkungsweise eines Schrittmotors mit Impulsdiagramm für den Rechts- und Linkslauf**




**Ansteuerschaltung für einen Schrittmotor durch den NASCOM 1, wobei die Datenleitungen durch Optokoppler galvanisch von der Leistungselektronik getrennt sind**

## Die Elektronik

Zum besseren Verständnis der elektrischen Schaltung sei kurz die Wirkungsweise der hier angewandten Gegentakt-Zweistrang-Ansteuerung von Schrittmotoren beschrieben:

Im Schrittmotor werden durch zwei feststehende Elektromagnete zwei Magnetfelder erzeugt, die um den Schrittwinkel verdreht sind (im Schema 90°). Der Rotor, der einen Permanentmagneten trägt, stellt sich entsprechend dem Gesamtmagnetfeld ein und wird so festgehalten. Wenn man in einer Spule die Stromrichtung und somit auch das dazugehörende Magnetfeld umpolt, dann ist das Gesamtmagnetfeld um 90° verdreht, wodurch sich der Rotor um denselben Betrag dreht, um wieder eine stabile Stellung zu erreichen. Wenn man nun den zweiten und dann wieder den ersten Magneten umpolt usw., erreicht man ein schrittweises Weiterdrehen des Rotors. Die Phasenverschiebung (+ oder − 90°) der beiden Magnetfelder bzw. Spulenströme zueinander bestimmt dabei die Drehrichtung des Motors.

Die Steuersignale für die Schrittmotoren werden über den Parallel-Input/Output des NASCOM ausgegeben und mittels der Optokoppler galvanisch getrennt in die Treiberschaltung übertragen. Die folgenden vier identischen Teilschaltungen steuern die einzelnen Wicklungen der Motoren so an, daß sie abwechselnd an +46 V und −46 V gelegt werden. Die Schutzdioden parallel zu den Wicklungen verhindern eine mögliche Zerstörung der Transistoren durch induktive Spannungsspitzen. Der Elektromagnet zum Anheben des Zeichenstiftes wird in ähnlicher Weise über den fünften Kanal angesteuert. Da die Treiberschaltungen galvanisch vom Mikrocomputer getrennt sind, wird noch eine zusätzliche 5-V-Spannung zur Versorgung der TTL-Bausteine benötigt. Das Gnd-Potential dieser Spannung ist mit der negativen Spannung für die Motoren verbunden.

Die 5-V-Spannung wird am besten mit einer Spannungsstabilisierung geregelt; für die Versorgungsspannungen der Motoren genügt je ein Elektrolytkondensator nach der Gleichrichtung, um die ±48 V ausreichend zu glätten.

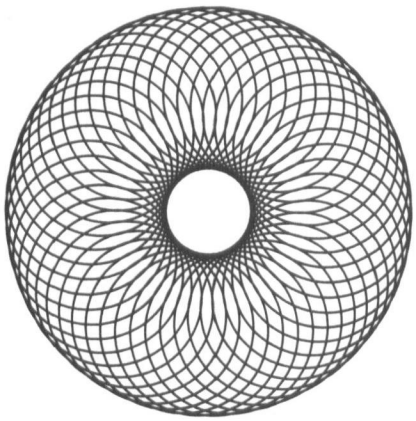
## Die Software

Der Zustand eines Ausgabebit an der Schnittstelle des Mikrocomputers bestimmt die Stromrichtung in einer Wicklung eines Motors. Soll der Motor schrittweise gedreht werden, so müssen für einen Motor folgende Bitkombinationen ausgegeben werden:

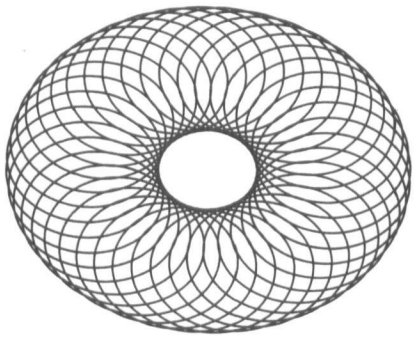
Schritt 1: 11
Schritt 2: 10
Schritt 3: 00
Schritt 4: 01

Diese Bitmuster kann man erhalten, wenn man das 8-bit-Wort 00110011 nach links rotiert und dabei die zwei rechten Bit beobachtet und an die Interface-Schaltung zum Motor weitergibt.

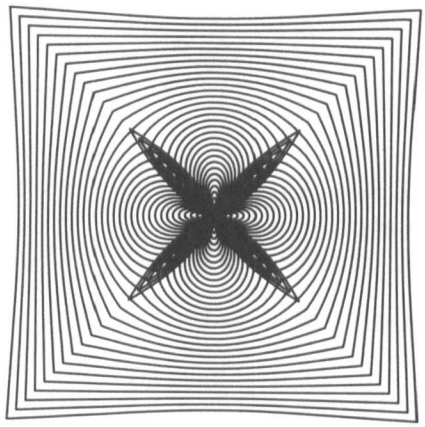
00110011 Schritt 1
01100110 Schritt 2
11001100 Schritt 3
10011001 Schritt 4
00110011 Schritt 5
identisch mit Schritt 1



**Kreisform durch den Selbstbauplotter gezeichnet**

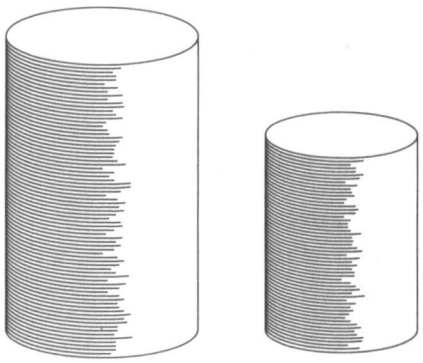


**Veränderte Kreisform durch die Software erzeugt**



**Linienmuster und Strukturen werden durch die Software erzeugt und durch den Selbstbauplotter gezeichnet**

**Mittels der Software lassen sich Kreise auch anders darstellen, wobei dann der Fantasie keine Grenzen gesetzt sind.**



**Darstellungen von Tonnen, gezeichnet und schraffiert durch die Software im Computer, dargestellt durch den Selbstbauplotter**

An diesem Beispiel ist zu sehen, daß die absolute Positionierung nur über einen Bereich von vier Schritten erfolgen kann, denn bereits beim fünften Schritt wiederholt sich das Bitmuster.
Selbstverständlich möchte man im Mikrocomputer eine ständige Angabe über die absolute Position des Motors bzw. des Zeichenstiftes über einen viel größeren Bereich haben. Dazu muß einfach mit jedem Rechts- oder Linksschritt ein Positionszähler, der als 16-bit-Wort im RAM-Bereich abgelegt ist, hoch- oder heruntergezählt werden. Um Schrittfehler zu vermeiden, muß der Mikrocomputer nach jedem neuen Schrittbefehl eine Pause machen, damit der Motor genügend Zeit hat, seine Position zu erreichen, bevor der nächste Schrittbefehl ausgegeben wird.
Für die zwei Motoren des Plotters, die ja unabhängig voneinander angesteuert werden sollen, ist das Programm schon etwas komplizierter. Wenn man nur waagerechte oder senkrechte Linien zeichnen will, genügt es, wenn sich nur je ein Motor dreht. Um aber beliebige schräge Linien zeichnen zu können, wird etwas Arithmetik benötigt, damit alle Drehzahlverhältnisse der zwei Motoren möglich sind.
Dazu müssen zuerst aus den Absolutkoordinaten von Anfang und Ende der zu zeichnenden Linie die Relativbewegungen in *X*- und *Y*-Richtung bestimmt und je nach deren Vorzeichen der Drehsinn der Motoren festgelegt werden. Ist die Bewegung für einen Motor gleich Null, wird nur der andere um die entsprechende Schrittzahl bewegt. Mit einer Division von ΔY durch ΔX wird die Steigung der zu zeichnenden Linie berechnet. Dann wird Schritt um Schritt eine Treppenkurve gezeichnet, die der theoretischen exakten Linie möglichst nahe kommt. Dies geschieht so, daß das Programm zunächst einen waagerechten Schritt ausführt und darauf solange in der Senkrechten weiterzeichnet, bis die Treppenkurve die theoretische Linie berührt oder kreuzt. Dann wird wieder ein waagerechter Schritt gezeichnet und dieser Algorithmus so lange wiederholt, bis der Zielpunkt erreicht ist.
Das Plottersteuerprogramm für den NASCOM-1 belegt in einem 2708-EPROM den Speicherplatz von hexadezimal CCØØ bis CFFF und für die Ansteuerung mit BASIC und für das Programm zur Kompensation des mechanischen Spielraumes noch einen weiteren EPROM-Speicherbereich von 8FØØ bis 8F73. Für die Argumentübergabe, die Positionszähler usw. wird noch der Speicherplatz im RAM-Bereich von ØC5Ø bis ØC7F benötigt. Für die Datenausgabe ist der Port Ø5 des NASCOM vorgesehen.

Das Programm ist so gestaltet, daß der Plotter auf drei verschiedene Arten gesteuert werden kann:

- mit direkten Befehlen
- mit Maschinenkode-Programmen
- mit BASIC-Programmen (8K-ROM-Version)

## Start bzw. Initiieren des Programmes

Im Direktmodus erfolgt der Kaltstart mit E CCØØ. Dadurch wird der RAM-Bereich mit den notwendigen Daten geladen, die Koordinaten auf Null gesetzt, die PIO aktiviert und eine neue Kommandotabelle erstellt. Danach kann der NASCOM außer den Plotterkommandos noch die Execute, Modify und Tabulate-Befehle ausführen. Ein Warmstart ist durch E CC16 möglich, dabei werden die bereits vorhandenen Koordinaten beibehalten. Für die Ansteuerung im Maschinenkode kann die Initiierung vor dem Start des aufrufenden Programmes wie oben erfolgen, oder durch Aufruf des Unterprogrammes bei 8F35. In BASIC muß vor dem ersten Plotteraufruf mit „DOKE 41ØØ – 28852" die Adresse des

Interfaceprogrammes an den BASIC-Interpreter übergeben werden. Mit USR(Ø) wird dann das Programm initiiert, was z.B. mit der Anweisung „Q = USR(Ø)" erfolgen kann.

## Beschreibung der Kommandos und Unterprogramme

### Heben des Zeichenstiftes

Im Direktmodus gibt man zu Beginn einer neuen Zeile den Buchstaben H und „New Line" ein und dann zieht der Hubmagnet den Zeichenstift hoch. Das dazugehörende Programm beginnt bei CD97 und kann im Maschinenkode als Unterprogramm aufgerufen werden. In BASIC läßt sich der Hubvorgang mit USR(1) auslösen.

### Senken des Zeichenstiftes

Dies erfolgt in Analogie zum Heben mit dem Kommando S. Das Unterprogramm dazu beginnt bei CDAØ und kann in BASIC mit USR(2) aufgerufen werden.

### Nullsetzen der Koordinatenwerte

Will man den augenblicklichen Stand des Zeichenstiftes weiterhin als neuen Koordinatenursprung, so kann das mit dem Kommando Z (wie Zero) im Direktmodus oder durch Aufruf des Unterprogrammes bei CC2D oder in BASIC mit USR(5) erfolgen.

### Absolute Positionierung

Will man im Direktmodus den Zeichenstift an eine bestimmte Stelle positionieren, so gibt man den Buchstaben A, gefolgt von zwei Dezimalzahlen (im Bereich von −32768 bis +32767), die die *X*- und *Y*-Koordinaten bezeichnen, ein, und nach „New Line" werden diese beiden Zahlen auf dem Bildschirm des NASCOM in der obersten Zeile rechts angezeigt. Gleichzeitig laufen die beiden Schrittmotoren an und deren Koordinaten werden auf dem Bildschirm links oben laufend als Hexadezimalzahlen angezeigt. Wenn die Motoren zum Stillstand gekommen sind, können wieder neue Kommandos eingegeben werden. Es ist darauf zu achten, daß die eingegebenen Zahlenwerte stets innerhalb der Zeichenfläche bleiben, denn eine automatische Abschaltung bei Bereichsüberschreitung des Zeichenstiftes ist bei diesem einfachen Gerät noch nicht vorgesehen.
Mit einem Maschinenkode-Programm läßt sich der Zeichenstift durch Aufruf des Unterprogrammes bei CF5B absolut positionieren. Vorher müssen jedoch die Argumente als 16-bit-Binärzahlen (höchstwertiges Bit ist Vorzeichenbit) übergeben werden. Das X-Argument muß in die Speicherplätze ØC54 und ØC55, das Y-Argument in die Plätze ØC56 und ØC57 geladen werden.
In BASIC erfolgt die Datenübergabe mit „DOKE 3156,X" und „DOKE 3158,Y", der Aufruf des Zeichenvorganges mit USR(3). Mit dem Unterprogramm bei 8FØØ bzw. mit USR(6) läßt sich der Zeichenstift so positionieren, daß, wie bereits beschrieben, das mechanische Spiel kompensiert wird.

### Relative Positionierung

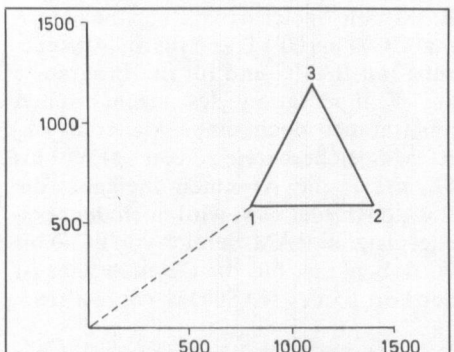
Um nacheinander beliebige Streckenzüge wie bei einer Vektoraddition ohne Rücksicht auf den Koordinatenursprung zu zeichnen, kann man sich dieser relativen Positionierung bedienen. Die Dateneingabe im Direktmodus erfolgt mit dem Buchstaben R und den relativen Koordinaten als Dezimalzahlen wie beim absoluten Positionieren. Dieses Verfahren kann ebenso mit dem Unterprogramm bei CF75 oder in BASIC mit USR(4) aufgerufen werden, wobei wie oben beschrieben die Daten vorher übergeben werden müssen.

### *X*/*Y*-Verschiebung

Dieses Kommando bewirkt eine Koordinatentranslation und dient dazu, den Zeichenstift bzw. die Trommel zu bewegen, ohne den Absolutpositionszeiger in seinem Betrag zu verändern. Im Direktmodus folgt nach den Buchstaben X oder Y eine entsprechende Dezimalzahl. Beim Aufruf dieser Verschiebung als Unterprogramm (CF14 für X und CF34 für die *Y*-Richtung) muß das betreffende Argument im Register HL stehen. In BASIC kann dieses Kommando nicht aufgerufen werden.

### Beispiel eines Zeichenvorganges

Folgendes Dreieck sei zu zeichnen:




**Grundplotter zur Erstellung der Software**

Im Direktmodus sieht die Kommandofolge so aus:

```
Z              Die augenblickli-
               che Position des
               Zeichenstiftes
               sei in Zukunft der
               Koordinatenur-
               sprung.
H              Der Stift wird an-
               gehoben.
A 8ØØ 6ØØ      Der Stift wird an
               Punkt 1 gefahren
               und abgesenkt.
S
A 14ØØ 6ØØ     (oder R 6ØØ Ø)
               Linie zu Punkt 2
A 11ØØ 12ØØ    (oder R -3ØØ 6ØØ)
               Linie zu Punkt 3
A 8ØØ 6ØØ      (oder R-3ØØ -6ØØ)
               Linie zu Punkt 1
H
A 12ØØ         Rückkehr zum Ur-
               sprung
S
```

```
BASIC-Programm zum Zeichnen
des Dreieckes:
DOKE 41ØØ,-28852
Q = USR(Ø)
DOKE 3156,8ØØ : DOKE 3158,6ØØ
Q = USR(1) + USR(6) + USR(2)
DOKE 3156,14ØØ : DOKE 3158,6ØØ
Q = USR(6)
DOKE 3156,11ØØ : DOKE
3158,12ØØ
Q = USR(6)
DOKE 3156,8ØØ : DOKE 3158,6ØØ
Q = USR(6)
DOKE 3156,Ø : DOKE 3158,Ø
Q = USR(1) + USR(6) + USR(2)
END
```

```
8f00 2a 54 0c ed 5b 56 0c e5  a8
8f08 d5 ed 5b 50 0c b7 ed 52  06
8f10 3a 5a 0c f5 cd 0f ce f1  cf
8f18 21 5a 0c be 28 0b fe 23  40
8f20 21 12 00 cc 28 ce cd 14  85
8f28 cf d1 e1 22 54 0c ed 53  fa
8f30 56 0c c3 5b cf 21 1f cc  1a
8f38 11 58 0c 01 0e 00 ed b0  e8
8f40 cd 2d cc 3e 0f d3 07 3e  fa
8f48 ff d3 05 c9 cd 8b e9 7b  33
8f50 fe 00 ca 35 8f fe 01 ca  34
8f58 97 cd fe 02 ca a0 cd fe  80
8f60 03 ca 5b cf fe 04 ca 75  27
8f68 cf fe 05 ca 2d cc fe 06  90
8f70 ca 00 8f c9 00 00 00 00  21
```

```
cc00 21 1f cc 11 58 0c 01 0e  5c
cc08 00 ed b0 cd 2d cc 3e 0f  84
cc10 d3 07 3e ff d3 05 21 de  ca
cc18 cf 22 45 0c c3 86 02 33  a4
cc20 33 23 c9 23 c9 cb 06 c9  91
cc28 cb 06 c9 08 90 af 21 50  46
cc30 0c 06 08 77 23 10 fc cd  89
cc38 4e cd 2a 54 0c 11 e0 0b  a5
cc40 cd f5 cc 2a 56 0c c3 f5  de
cc48 cc 1a fe 20 13 28 fa 1b  68
cc50 1a fe 2d 20 03 13 18 07  b6
cc58 fe 2b 20 01 13 3e 2b 32  1c
cc60 6c 0c 1a fe 20 13 28 fa  11
cc68 1b af 21 66 0c 06 06 77  14
cc70 23 10 fc 1a d6 30 f8 fe  81
cc78 0a f0 d5 21 6a 0c 11 6b  26
```

```
cc80 0c 01 04 00 ed b3 12 34 48
cc88 d1 13 13 e7 26 00 cd 9a c4
cc90 cc ed 5b 6d 0c 19 22 6d 91
cc98 0c c9 f5 c5 06 10 4a 7b ce
cca0 eb 21 00 00 cb 39 1f 30 cb
cca8 01 19 eb 29 eb 10 f5 c1 53
ccb0 f1 c9 d5 26 00 3a 67 0c de
ccb8 6f 22 6d 0c 3a 68 0c 6f ab
ccc0 11 0a 00 cd 8c cc 3a 69 6f
ccc8 0c 6f 11 64 00 cd 8c cc a9
ccd0 3a 6a 0c 6f 11 e8 03 cd 34
ccd8 3c cc 3a 6b 0c 6f 11 10 3d
cce0 27 cd 8c cc d1 3a 6c 0c 7b
cce8 fe 2d c0 2a 6d 0c cd 28 37
ccf0 ce 22 6d 0c c9 f5 c5 e5 8d
ccf8 d5 cb 7c 3e 20 28 05 cd 38
cd00 28 ce 3e 2d 12 13 01 10 64
cd08 27 cd 41 cd 12 13 01 e8 e5
cd10 03 cd 41 cd 12 13 01 64 45
cd18 00 cd 41 cd 12 13 01 0a f0
cd20 00 cd 41 cd 12 13 7d c6 30
cd28 30 12 13 13 13 d9 d1 06 20
cd30 04 13 1a fe 30 20 05 3e bf
cd38 20 12 10 f5 d9 e1 c1 f1 a8
cd40 c9 af 3c b7 ed 42 f2 42 db
cd48 cd 09 3d c6 30 c9 11 d0 c8
cd50 0b 2a 50 0c cd 5b cd 2a cd
cd58 52 0c 13 7c cd 60 cd 7d 89
cd60 f5 1f 1f 1f 1f cd 69 cd a1
cd68 f1 e6 0f c6 30 fe 3a 38 31
cd70 02 c6 07 12 13 c9 db 05 da
cd78 47 3e f0 a0 47 21 58 0c 26
cd80 3e 03 a6 b0 47 23 3e 0c 98
cd88 a6 b0 d3 05 3a 64 0c f5 22
cd90 f1 f5 f1 3d 20 f9 c9 cb 1e
cd98 a0 cd 79 cd 3e 20 18 07 95
cda0 cb e0 cd 79 cd 3e 40 f5 9e

cda8 af f5 f1 f5 f1 3d 20 f9 46
cdb0 f1 3d 20 f3 c9 21 58 0c 0c
cdb8 cd 5e 0c 2a 70 0c 2b 22 af
cdc0 70 0c 2a 50 0c cd 5a 0c c2
cdc8 22 50 0c cd 4e cd c3 76 34
cdd0 cd 21 59 0c cd 61 0c 2a 54
cdd8 72 0c 2b 22 72 0c 2a 52 6a
cde0 0c cd 5c 0c 22 52 0c 18 86
cde8 e1 2a 72 0c 7c b5 c8 cd 04
cdf0 d1 cd 3a 65 0c f5 f1 3d 29
cdf8 20 fb 18 ed 2a 70 0c 7c 07
ce00 b5 c8 cd b5 cd 3a 65 0c 45
ce08 f5 f1 3d 20 fb 18 ed cb e4
ce10 7c 20 0b 3e 23 32 5a 0c 7e
ce18 3e 06 32 5f 0c c9 3e 2b f9
ce20 32 5a 0c 3e 0e 32 5f 0c 6f
ce28 7d 2f 6f 7c 2f 67 23 c9 0f
ce30 06 1f 21 00 00 22 7c 0c ee
ce38 21 78 0c cd 68 ce 30 14 f2
ce40 ed 5b 70 0c 2a 7c 0c 19 9d
ce48 22 7c 0c 30 07 2a 7e 0c ab
ce50 23 22 7e 0c 21 7c 0c cd 63
ce58 68 ce 10 dc 21 7d 0c cb bd
ce60 7e 2a 7e 0c 28 01 23 c9 75
ce68 cb 26 23 cb 16 23 cb 16 2f
ce70 23 cb 16 c9 f5 c5 af 32 a6
ce78 74 0c 32 75 0c 0e 11 cb 63
ce80 72 20 07 cb 23 cb 12 0c be
ce88 18 f5 b7 ed 52 f2 97 ce b0
ce90 19 e5 21 74 0c 18 06 e5 00
ce98 21 74 0c cb c6 0d 23 07 d4
cea0 cd 68 ce e1 29 18 e3 e1 57
cea8 2a 74 0c ed 5b 76 0c c1 ab
ceb0 f1 c9 11 4c 0b cd 49 cc 82
ceb8 3a 66 0c b7 c9 cd b2 ce ff
cec0 20 0b 2a 54 0c 22 70 0c e1
cec8 2a 56 0c 18 12 cd b2 cc 97
ced0 22 70 0c 22 54 0c cd 49 d4

ced8 cc cd b2 cc 22 56 0c 22 63
cee0 72 0c c3 75 cf cd 28 ce f6
cee8 22 70 0c 22 54 0c 21 00 f7
cef0 00 18 e9 cd 28 ce 22 72 16
cef8 0c 22 56 0c 21 00 00 22 99
cf00 54 0c 22 70 0c 18 db cd 8d
cf08 b2 ce 20 05 2a 54 0c 18 1e
cf10 06 cd b2 cc 22 54 0c 22 d4
cf18 70 0c ed 5b 50 0c eb b7 a9
cf20 ed 52 22 50 0c 18 c7 cd 58
cf28 b2 ce 20 05 2a 56 0c 18 40
cf30 06 cd b2 cc 22 56 0c 22 f6
cf38 72 0c ed 5b 52 0c eb b7 cd
cf40 ed 52 22 52 0c 18 b5 cd 68
cf48 b2 ce 28 0f cd b2 cc 22 3b
cf50 54 0c cd 49 cc cd b2 cc ac
cf58 22 56 0c 2a 54 0c ed 5b 7d
cf60 50 0c b7 ed 52 22 70 0c 1f
cf68 2a 56 0c ed 5b 52 0c b7 20
cf70 ed 52 22 72 0c 2a 72 0c c6
cf78 cd 0f ce 3a 5a 0c 32 5c 1f
cf80 0c 3a 5f 0c 32 62 0c 22 c2
cf88 72 0c 2a 70 0c cd 0f ce 25
cf90 22 70 0c cd 3a cc 2a 70 6a
cf98 0c 7c b5 ca e9 cd 2a 72 c0
cfa0 0c 7c b5 ca fc cd 2a 72 db
cfa8 0c ed 5b 70 0c cd 74 ce 56
cfb0 cd b5 cd 21 74 0c 11 78 f8
cfb8 0c 01 04 00 ed b0 cd 30 32
cfc0 ce ed 5b 72 0c b7 ed 52 19
cfc8 28 08 f4 b5 cd fc d1 cd d7
cfd0 18 e1 21 70 0c 7e 23 b6 8c
cfd8 c8 c4 b5 cd 18 d5 59 27 22
cfe0 cf 58 07 cf 48 97 cd 53 ab
cfe8 a0 cd 41 47 cf 52 bd ce 58
cff0 5a 2d cc 4d ad 01 54 fc 5d
cff8 01 45 d0 02 00 3d 00 00 1c
```

# Speicher auslesen mit druckendem Kaufhausrechner

*Fortsetzung von Seite 124*

**Kompletter Aufbau zwischen Introkit (SC/MP) und dem druckenden Rechner**

nen sich am besten ein paar Optokoppler mit vorgeschalteten Transistorstufen. Die gesamte Interface-Schaltung ist also recht einfach und kann in einer halben Stunde zusammengebaut werden. Jetzt fehlt nur noch eine Druckroutine, um das ganze in Gang zu setzen.

## Das Programm

Das Druck-Programm hat die Aufgabe, ab einer vorgegebenen Anfangsadresse eines Speicherbereiches den Inhalt der einzelnen Adressen der Reihe nach zu holen, umzuformen und über die Flags dem Drucker zuzuführen.

Das Listing des Druck-Programmes beginnt mit der Festlegung der Anfangsadresse, hier z.B. OF 90, der Zeilenanzahl (OC) und der Anzahl der Zeichen pro Zeile (08). Es folgt die Ladung und Zwischenspeicherung der Werte. Das erste Bit des Wertes wird dann isoliert, auf seinen Inhalt überprüft und über das Statusregister ausgegeben.

Eine „1" im abgefragten Bit bewirkt die Ansteuerung von Flag 1; eine „0" steuert Flag 2 an. Dieser Vorgang wird so oft wiederholt, bis alle Zeichen einer Zeile in den Drucker eingelesen sind. Nach jeweils acht eingelesenen Zeichen leitet das Programm einen Druckvorgang ein, um eine Zeile auszugeben. Diese Prozedur wird so lange fortgesetzt, bis die Inhalte aller Adressen ausgedruckt sind. Weitere Einzelheiten gehen aus dem Programmablaufplan und der Programmauflistung hervor.

## Schlußbetrachtung

Sicher kann eine solche Lösung kein Listing im Hexkode ersetzen. Für Mikrocomputer-Freunde mit kleinem Geldbeutel könnte es jedoch ein interessanter Behelf sein. Und wenn ein teureres Gerät nach einiger Zeit angeschafft werden kann, dann genügt es, die Verbindungsstrippen einfach wieder abzulöten, und der Drucker kann wieder im Büro verwendet werden.

Frank Wähner

# Akustik-Koppler

Zweck: Diese höchst einfache Schaltung dient dazu, Tonsignale vom Computer auszugeben bzw. zu empfangen. Genauigkeit des A/D- bzw. D/A-Umsetzers: 1 bit

## Baubeschreibung

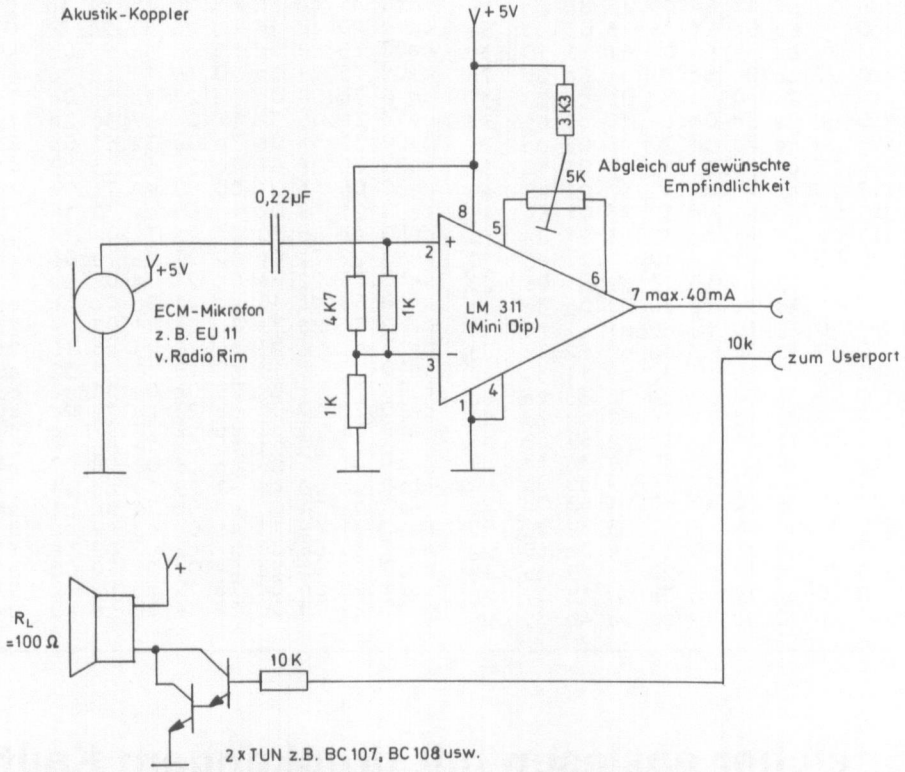
Der Aufbau ist absolut problemlos und wird am besten auf einer Veroboard-Platine vorgenommen.

## Abgleich und Test der Funktion

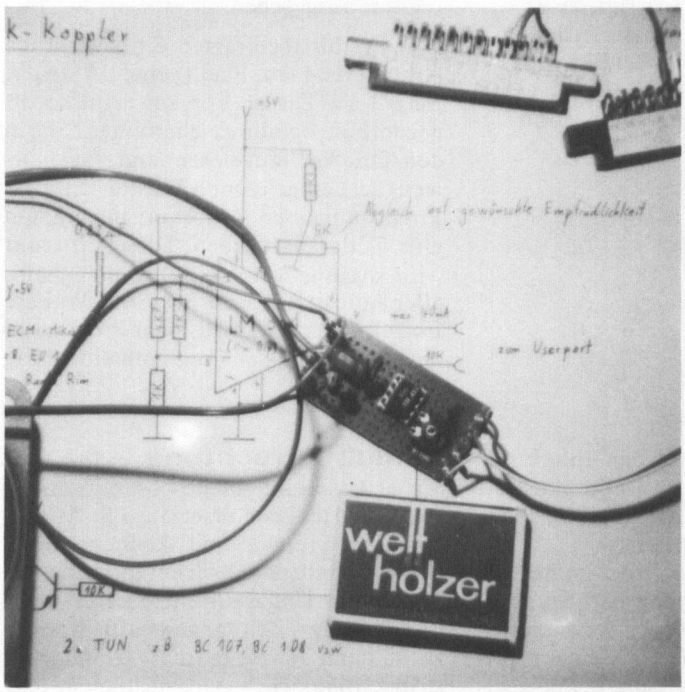
Lautsprecher (100 Ω) wird an den Ausgang des Komparators LM 311 angeschlossen (der andere Lautsprecher wird mit +5 V verbunden). Wird nun das Mikrofon in die Nähe des Lautsprechers gebracht, kann man beim gleichzeitigen Drehen des Trimmers ein lautes Rückkoppelungspfeifen vernehmen. Andernfalls wurde die Schaltung nicht korrekt aufgebaut.

## Anschluß an einen Computer

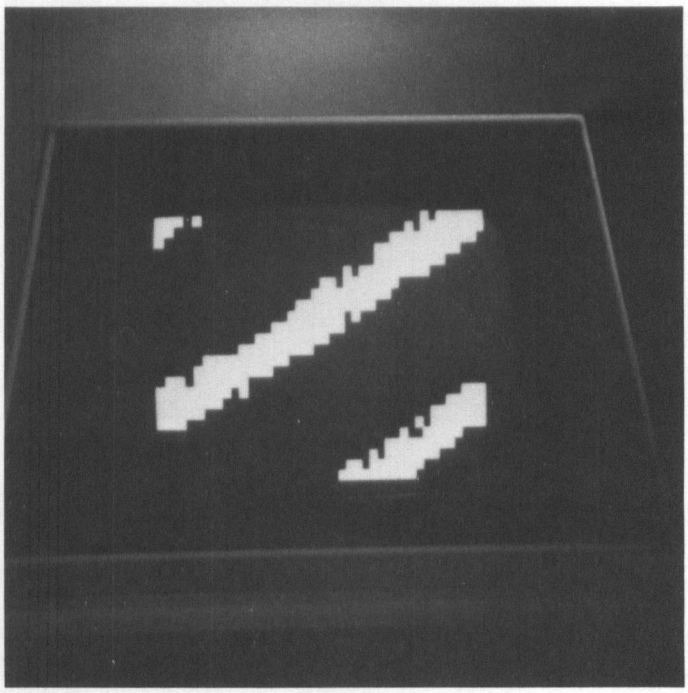
Dank des geringen Stromverbrauchs kann die Schaltung direkt an die 5 V des Computers angeschlossen werden.




**Schaltung für den Akustik-Koppler**




**Platine für den Akustik-Koppler**



**So „hört" der CBM 2001 die Kennmelodie von Bayern III vor einer Verkehrsdurchsage**

**Beispielprogramm für den Akustik-Koppler für CBM 2001 („Lichtorgel"):**

```
START    JSR BREAK
         LDA U
         AND #1
         BEQ START     ; USERPORT AUF LOGISCH 0 ?
         LDX #255
L1       DEX
         BNE L1        ; ZEITVERZÖGERUNG
         JSR BREAK
         LDA U
         AND #1
         BEQ START     ; KANN ABSPEICHERUNG BEGINNEN ?
         LDA #0
         STA 1
         LDA #128
         STA 2         ; INITIALISIEREN DES ZEIGERS
         SEI
         LDY #0
L2       LDX #32
         LDA U
         AND #1
         BEQ L3
         LDX #160
L3       TXA
         STA (1),Y
         INY
         BNE L2        ; UEBERTRAG ?
         INC 2
         LDA 2
         CMP #132
         BNE L2        ; FERTIG ?
         CLI
         LDA 516
         BEQ START     ; ZWISCHENSPEICHERN ?
L4       LDA 515
         CMP #255
         BEQ L4        ; TASTE GEDRÜCKT ?
         CMP #4
         BNE START     ; STOP-TASTE GEDRÜCKT ?
RETURN   RTS
BREAK    LDA 515
         CMP #4
         BNE RETURN    ; ABBRUCH ?
         JMP 65505

U      : 59457         ; USERPORT (PORT B VON VIA #1)
```

Der Ausgang des Komparators wird direkt an PAØ vom Userport (Pin C), der Eingang des Verstärkers an PA1 vom Userport (Pin D) angeschlossen (Anschlußbelegung von CBM).

## Eingabe des Beispielprogramms

Ist zum Beispiel mit meinem Assembler ein Kinderspiel. Startadresse beliebig, z.B. 7800. Das Listing gilt für den CBM 2001. Bei der 30XX- bzw. 40XX-Serie müssen lediglich die Adressen 515, 516 gegen 151, 152 ausgetauscht werden.

## Bedienung des Programms

Nach dem Starten erscheinen auf dem Bildschirm je nach Lautstärke und Frequenz Muster auf dem Bildschirm. Eventuell ist eine kleine Korrektur der Trimmereinstellung nötig. Wird nun die Shift-Taste gedrückt, so erstarrt die Anzeige und reagiert auf keine Geräusche. Nach dem Drücken einer beliebigen Taste wird die Anzeige fortgesetzt. Unterbrochen wird das Programm durch die Run/Stop-Taste.

Wolfram Bahmann

# Low-cost-Drucker für EUROCOM 1

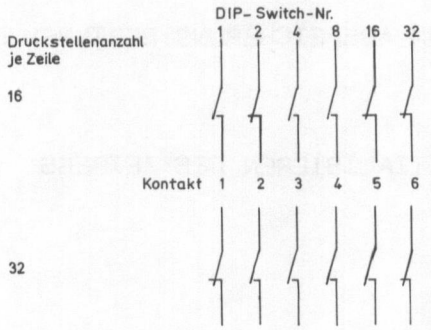
Nachfolgend wird die Ausrüstung eines Kleindrucksystems mit einer softwaregesteuerten Umschaltung der Druckstellenanzahl (32/16 Zeichen je Zeile) sowie dessen Anschluß an den EUROCOM 1 beschrieben und die Software für ein komfortables Speicher-Listing (Hex-Dump) dargelegt.

Auch beim Umgang mit kleinen Mikroprozessor-Systemen besteht meist der Wunsch nach einer Möglichkeit, Speicherinhalte und/oder Ergebnisse von Operationen direkt sichtbar zu dokumentieren, d.h. einen Druckstreifen zu erzeugen. Eine der preiswertesten Produktmöglichkeiten, die auch gleichzeitig sehr platzsparend ist, stellt wohl der MINI-DOT-PRINTER von LOGITEC/ München mit knapp 400 DM dar. Er besteht aus einem elektromechanischen Teil und einer darunter montierten Parallel-Interface-Platine. Es handelt sich um einen Matrix-Drucker für metallisiertes Papier (60 mm breit).

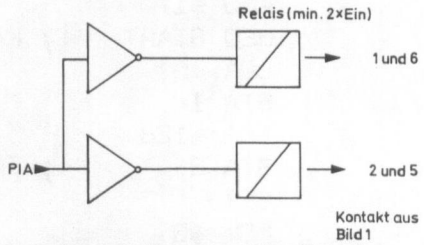
Einschließlich der weiter unten beschriebenen Ergänzung des Gerätes, die sonst nur über DIL-Schalter vorzuwählende Druckstellenanzahl je Zeile per Software zu ändern, ist nur ein Port (hier PB) des EUROCOM – PIA 2 zum Anschluß erforderlich. Das Anschlußschema wird in der Programmbeschreibung genannt. Die erwähnte Umstellung der Druckstellenanzahl je Zeile per Software wird möglich, wenn man die Funktion des in die Platine eingelöteten DIL-Schalters durch ein oder mehrere Relais übernimmt, die von einem Output-Pin des Ports angesteuert werden. In dem Bild sind die sechs Kontakte allgemein dargestellt zusammen mit den jeweils erforderlichen Stellungen. Da die mittleren beiden Kontakte in beiden Fällen die gleiche Stellung haben, werden nur vier Umschaltkontakte benötigt. Je nach Verfügbarkeit können verschiedene Kleinrelais verwendet werden. Im speziellen Fall waren Reed-Relais 4 × Ein vorhanden. Auch hiermit ist die Umschaltung realisierbar: man benutzt zwei Relais, die mit invertierten Signalen angesteuert werden.

Die beiden jeweils außenliegenden Kontakte werden durch die Relaiskontakte ersetzt. Die Verbindung zwischen Interface-Platine und Relais kann aus einem Stück Flachbandkabel bestehen. Will man den DIL-Schalter nicht auslöten, so sind alle Schalter (1 bis 32) in die Position „ON" zu bringen, die sich wider alle Erfahrung als diejenige herausstellte, bei der keine Verbindung zwischen den beiden Schalterpolen besteht.

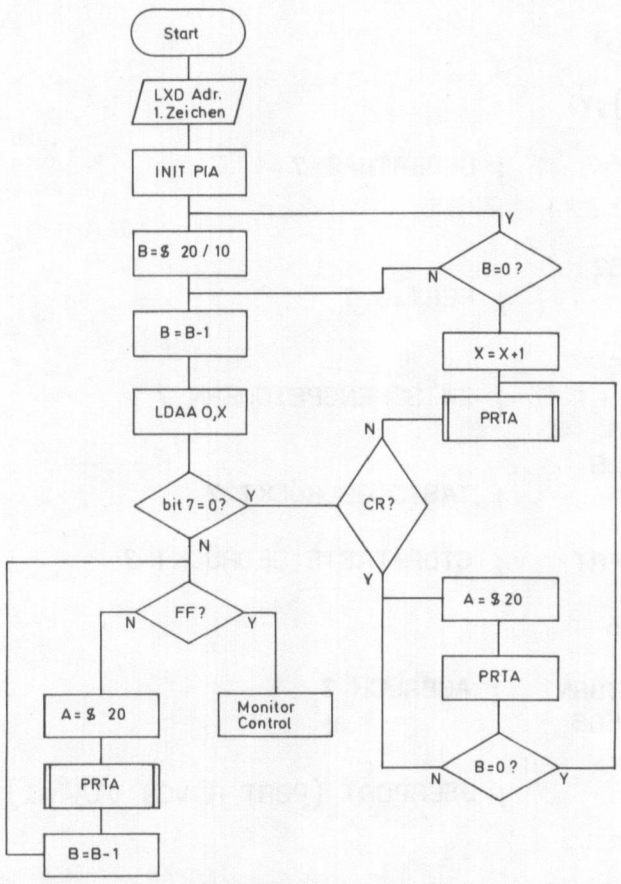
Zum Druckbetrieb selbst ist zu erwähnen, daß die Betriebsspannung für Motor und Schreibnadeln von −24 V eine meines Erachtens unzureichende Schwärzung des Metallpapiers bewirkt. Die Druckqualität steigt, wenn man die Spannung auf −27 oder −28 V erhöht.



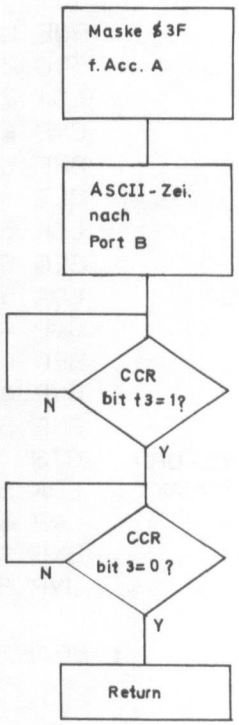

**Schalterstellungen für Drucker**



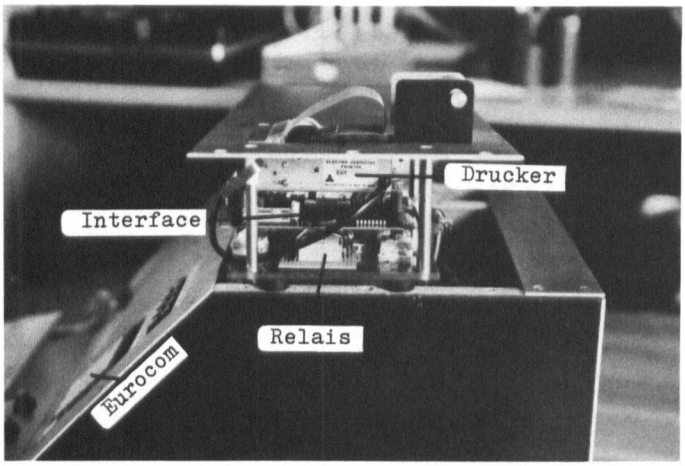

**Treiberschaltung für den Drucker**




**Programmablaufplan für Ausdruck eines ASCII-Zeichenfeldes**




**Unterprogramm PRTA**




**Geöffnetes Gehäuse des Computers und Bezeichnungen für den Drucker**

Ein Foto zeigt den in die Deckplatte eines Pultgehäuses eingebauten Drucker (D) mit den Umschaltrelais (R) in einer Seitenansicht. Im anderen Foto ist das gesamte Gehäuse mit EUROCOM 1 und ASCII-Tastatur zu erkennen. Letztere ist für die in Entwicklung befindliche Ausbaustufe gedacht, bei der auch ein Video-Interface adaptiert wird.

Als Anwendungsbeispiel wird nachfolgend ein Programm beschrieben, das einen HEX-Dump eines beliebig vorzuge-

```
MEMORY-HEX-DUMP

A400  4D 45 4D 4F 52 59 2D 48
A408  45 58 2D 44 55 4D 50 0D
A410  80 0D 20 5B 20 20 20 20
A418  20 5D 20 42 59 54 45 53
A420  0D 0D 0D 80 86 20 20 02
A428  86 10 97 00 DE 07 86 30
A430  B7 80 07 86 7F B7 80 05
A438  86 34 D6 00 C0 10 26 02
A440  86 3C B7 80 07 39 D6 00
A448  5A A6 00 2A 10 17 2B 07
A450  86 20 8D 1F 5A 20 F6 86
A458  FF B7 80 05 39 81 0D 27
A460  08 8D 10 08 17 27 DF 20
A468  DF 86 20 8D 06 17 27 F3
A470  5A 20 F6 84 3F B7 80 05
A478  7D 80 05 2B FB 7D 80 05
A480  2A FB 39 CE A4 00 DF 07
A488  8D 9E 8D BA 8D 96 4F 97
A490  0D 86 10 97 0E BD FC BA
A498  FE A7 C8 DF 01 DF 03 FE
A4A0  A7 CA DF 07 DE 0D DF 05
A4A8  C6 1E 96 03 8D 4B 96 03
A4B0  8D 59 96 04 8D 43 96 04
A4B8  8D 51 86 20 A7 00 08 A7
A4C0  00 08 DF 05 C0 06 DE 03
A4C8  A6 00 DE 05 8D 2B DF 05
A4D0  DE 03 A6 00 DE 05 8D 33
A4D8  86 20 A7 00 08 DF 05 DE
A4E0  03 9C 07 27 36 08 DF 03
A4E8  C0 03 26 DA DE 05 86 FF
A4F0  A7 00 DE 0D BD A4 46 20
A4F8  AB 44 44 44 44 81 09 2E
A500  04 8B 30 20 02 8B 37 A7
A508  00 08 39 84 0F 81 09 2E
A510  04 8B 30 20 02 8B 37 A7
A518  00 08 39 DE 05 86 FF A7
A520  00 DE 0D BD A4 46 96 08
A528  92 02 89 01 97 0C 96 07
A530  92 01 97 0B 8D 54 86 A4
A538  97 09 86 14 97 0A 7F 00
A540  37 86 33 97 38 7C 00 38
A548  86 34 91 38 27 0B DE 37
A550  A6 00 DE 09 BD A4 F9 DF
A558  09 DE 37 A6 00 DE 09 BD
A560  A5 0B DF 09 86 36 90 38
A568  26 DB 01 01 86 30 C6 20
A570  CE A4 14 A1 00 26 05 E7
A578  00 08 20 F7 CE A4 11 DF
A580  07 BD A4 28 BD A4 46 7E
A588  FA 54 4F 97 35 97 34 D6
A590  0C C4 0F 1B 19 97 36 CE
A598  A5 C4 D6 0C C4 F0 8D 05
A5A0  CE A5 D0 D6 0B 08 08 08
A5A8  58 24 02 8D 04 5D 26 F5
A5B0  39 96 36 AB 00 19 97 36
A5B8  96 35 A9 01 19 97 35 96
A5C0  34 A9 02 19 97 34 39 28
A5C8  01 00 64 00 00 32 00 00
A5D0  16 00 00 68 27 03 84 63
A5D8  01 92 81 00 96 40 00 48
A5E0  20 00 24 10 00 12 05 00
A5E8  56 02 00

[  491 ] BYTES
```

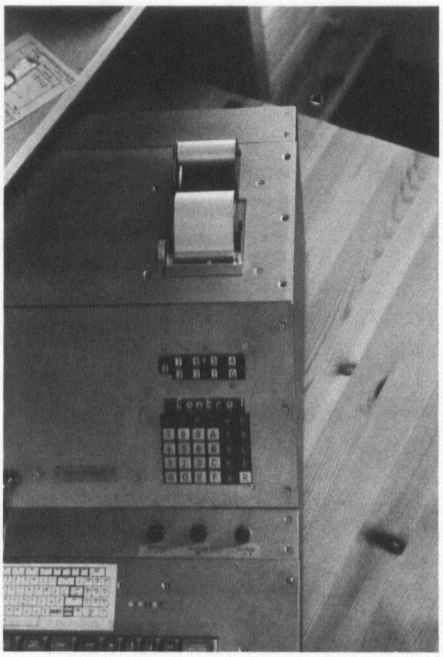
| Drucker-Interface an PIA 2, Port B | |
|---|---|
| Adr. 8005 = DDR/DR<br>Adr. 8007 = CR | |
| Interface (erweitert) | PIA |
| Data DO bis D5<br>Print Command ($\overline{PC}$)<br>Data Request (DR)<br>Relaistreiber | Bit 0 bis 5<br>Bit 6<br>Bit 7<br>CB 2 |

## Programmaufbau nach Funktionsblöcken

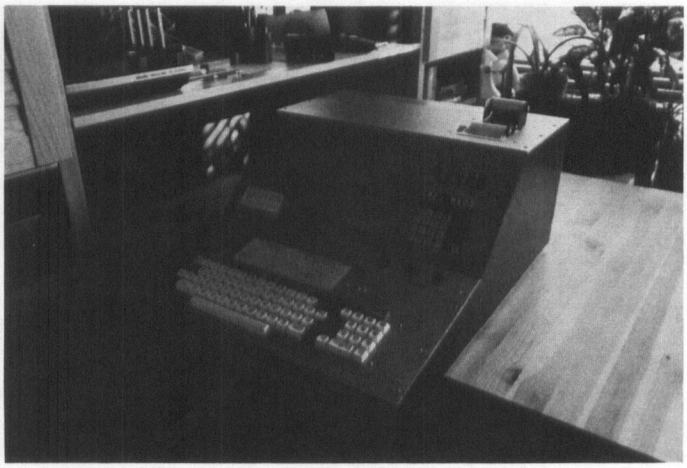
| Adresse | Bezeichnung |
|---|---|
| A400 | Text 1 (Anfang) |
| A411 | Text 2 (Ende) |
| A424/ | PIA-Initialisierung |
| A428 | 32/16 Zeichen je Zeile |
| A446 | Print Data |
| A483 | Print Text 1 (= START) |
| A48E | Hex-Dump |
| A526 | Berechnung der Anzahl der ausgedruckten Speicherstellen |
| A534 | Branch to Unterprogramm HEX-BCD |
| A536 | BCD-ASCII-Konvertierung |
| A57D | Print Text 2 |
| A585 | Return to Monitor |
| A58A | HEX-BCD-Konvertierung |
| A5EA = END | |

## Variablenspeicher

| Adresse nur Zero-Page | Bezeichnung |
|---|---|
| 00 | Druckstellenanzahl ($ 20 oder $ 10) |
| 01/02 | 1. Adr. des Dump-Bereiches |
| 03/04 | Ersatzadr. dto. |
| 05/06 | Data-Pointer |
| 07/08 | Endadr. d. Dump-Bereiches |
| 09/0A | Hilfsspeicher (BCD-ASCII) |
| 0B/0C | Speicherzellenanzahl des Dump-Bereiches |
| 0D/0E | 1. Adr. des Buffer-Speichers |
| 10 bis 33 | Buffer-Speicher (36 Byte) |
| 34/35/36 | 6stellige Dezimalzahl (HEX-BCD) |
| 37/38 | Zwischenspeicher (BCD-ASCII) |



**Kompletter Computer des Autors mit ASCII-Tastatur, ursprünglicher Tastatur und einem Drucker**



**EUROCOM 1 im Gehäuse mit Tastatur, Anzeige und einem preisgünstigen Drucker**

benden Speicherbereiches erzeugt (etwa 60 cm Papier je KByte).
Den Programmaufbau erläutert der Ablaufplan.

Nach Start mit G A483 erfolgt der Ausdruck „MEMO-HEX-DUMP" einschließlich einer Freizeile. Über die LED-Anzeige schließt sich die Frage nach dem Speicherbereich an: „bEG" und „End". Hierzu wird das Unterprogramm „AREA" des Monitors aufgerufen. Anschließend beginnt die Auflistung der Speicherinhalte, wobei am Zeilenanfang immer die jeweilige Adresse gedruckt wird. Eine Zeile enthält je acht Byte. Am Schluß des Ausdrucks wird nach einer Freizeile noch die dezimale Anzahl der Speicherzellen des gelisteten Bereiches angegeben:
„(123) BYTE". Die Umschaltung der Druckstellenanzahl von normalerweise 32 je Zeile auf 16 erfolgt für die erste und die letzte Zeile. Nebenstehend hat sich das Programm selbst aufgelistet.


# CHIP-Box.
## Die schöne Seite des Sammelns.

**CHIP.** Das Mikrocomputer-Fachmagazin bringt einfach alles, worauf es ankommt, wenn es um Mikrocomputer-Technik geht. Heft für Heft.

Man muß nicht alles wissen. Man muß aber wissen, wo es steht: Natürlich in der praktischen **CHIP-Box,** die das Nachschlagen, Sammeln und Aufbewahren der CHIP-Fachmagazine so angenehm und mühelos macht.
**CHIP-Sammel-Box.** Die muß man einfach haben.

**CHIP-Box.**

☐ Ja. Senden Sie mir die praktische CHIP-Sammel-Box für 12,– DM. Ausland 14,– DM, einschließlich Versandkosten. Ich bestelle _____ Stück.

Genaue Lieferanschrift:

Name, Vorname

Straße

PLZ/Ort

Datum/Unterschrift

CHIP-Leserservice, Vogel-Verlag, Postf. 67 40, D-8700 Würzburg 1

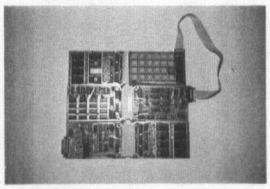
Fachbücher für Technik und Wirtschaft

Katalog liefert
VOGEL-VERLAG
Postfach 67 40
8700 Würzburg 1

**Technik, Wirtschaft, Universal**

Zahlreiche Titel der Fächer Mathematik, Physik, Chemie, Kunststoffe/-Verarbeitung, Elektrotechnik, Elektronik, Daten verarbeitung, Computertechnik, Maschinenbau, Kfz- und Landmaschinentechnik, Fertigungstechnik, Betriebsorganisation, Wirtschaft

**CHIP-SPECIAL-SELBSTBAU**



| | |
|---|---|
| Serielles Interface | 44,10 DM |
| Hex-Tastatur (2 Plat.) | 75,70 DM |
| ROM-Modul | 52,00 DM |
| Alle Preise inkl. MwSt. | |
| 8K-RAM (stat.) | 46,35 DM |
| CPU-Modul (8080A) | 31,65 DM |
| E/A-Modul (3 × 8255A) | 31,65 DM |
| Rückwand | 122,05 DM |

Weitere Platinen:
**Reinhold Ruf**
Oelingrath 1 a
**5630 Remscheid**

Klaus Gano

# CBM-Programmer

Das vorliegende Programmiergerät ist für CBM-Besitzer interessant, die ihre EPROM der Familie 2516 und 2716 selbst programmieren möchten.
Im Prinzip lassen sich Hard- und Software an alle gängigen EPROM-Typen anpassen.
Für die Programmierung von PROM werden, wie allgemein bekannt, acht Daten- und zehn Adressenleitungen sowie noch einige zusätzliche Leitungen für Programm- und Kontrollfunktionen benötigt.
Nun besitzt der CBM zwar einen User-Port, dessen acht Datenleitungen zum Datenaustausch geeignet sind, es fehlen jedoch die Adressenleitungen.
Dieses Problem läßt sich unter Ausnutzung der Tatsache lösen, daß die EPROM-Speicherzellen sequentiell in aufsteigender Reihenfolge adressiert werden, bis eine bestimmte Endadresse erreicht ist.

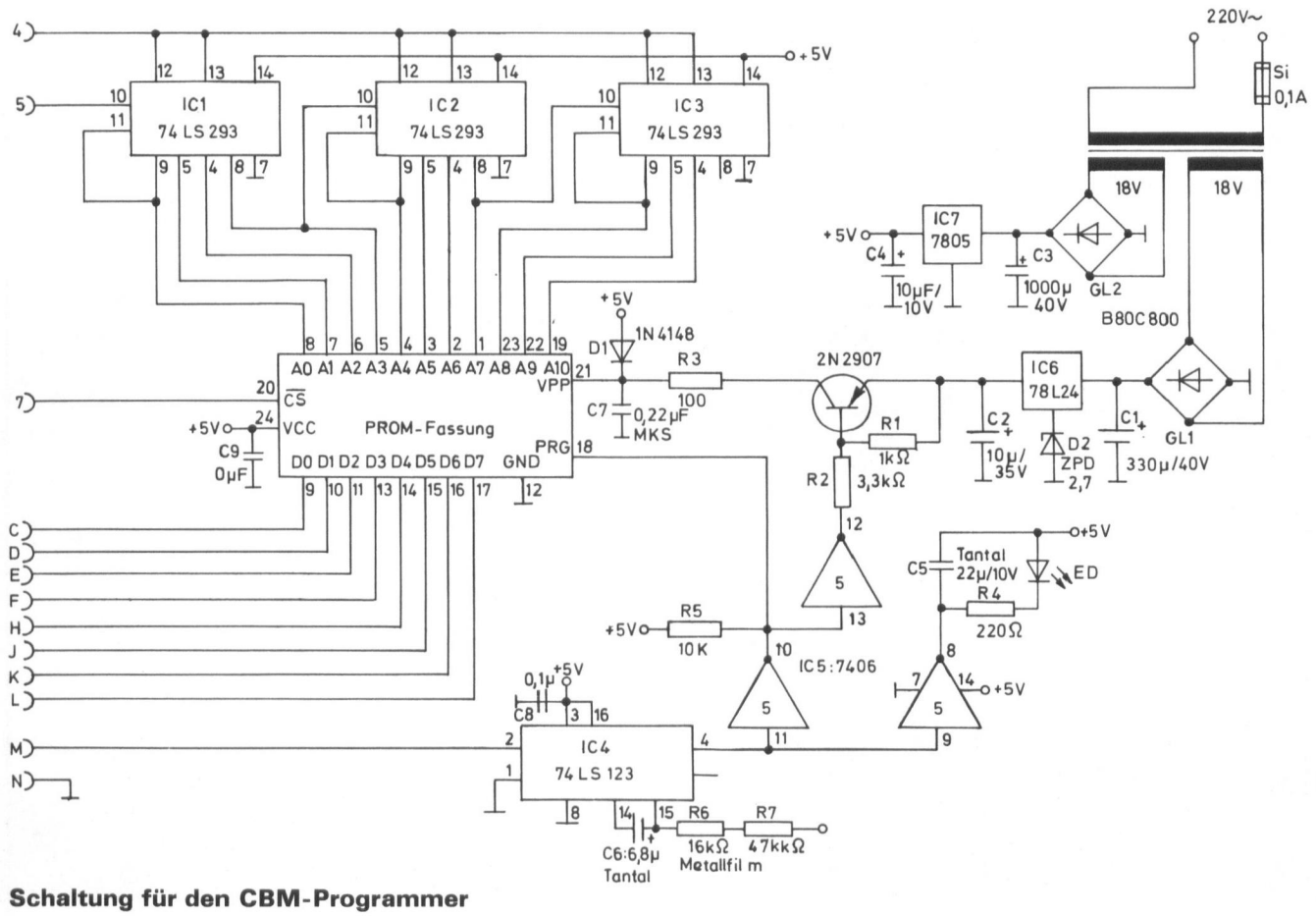
Man benötigt dazu einen Binärzähler, mit dessen Ausgängen die Adressenleitungen des EPROM gesteuert werden, sowie je eine Leitung für den Zähl- und Rücksetzimpuls, den $\overline{CS}$, der das Lesen von ROM-Daten ermöglicht, und einen Programmierimpuls mit einer Länge von 50 ms. Das Programm besitzt zwar eine Zeitschleife, es hat sich aber als zweckmäßig erwiesen, diesen Impuls extern mit einem Monoflop zu erzeugen. Sollte aus irgendwelchen Gründen ein Interrupt auftreten, so würde dieser Impuls länger als die erlaubte Zeit dauern und zur Beschädigung des EPROM führen.
Alle diese genannten Leitungen lassen sich aus den noch freien User-Port-Anschlüssen gewinnen, die sich noch als Ausgänge programmieren lassen.
Das Programm wurde ausschließlich in Maschinensprache erstellt und am Anfang des BASIC-Arbeitsspeichers abgelegt.

Zum Schutz vor Überschreibung mit BASIC wird bei Programmstart der Zeiger für BASIC verstellt (POKE 40,1 : POKE 41,14).
Der Vorteil des Programms liegt darin, daß sich jede gewünschte PROM-Adresse auch einzeln programmieren läßt.
Der RAM-Speicherbereich beginnt bei 1536, RAM-Startadresse-Lower Byte ist in 1395 und RAM-Startadresse-Higher Byte in 1396 abgelegt und endet für 2K bei 3583.

Die EPROM-Startadresse steht in 1393 (LB) und 1394 (HB).

Die EPROM- sowie die RAM-Endadresse sind in 1397 (LB) und 1398 (HB) gespeichert.

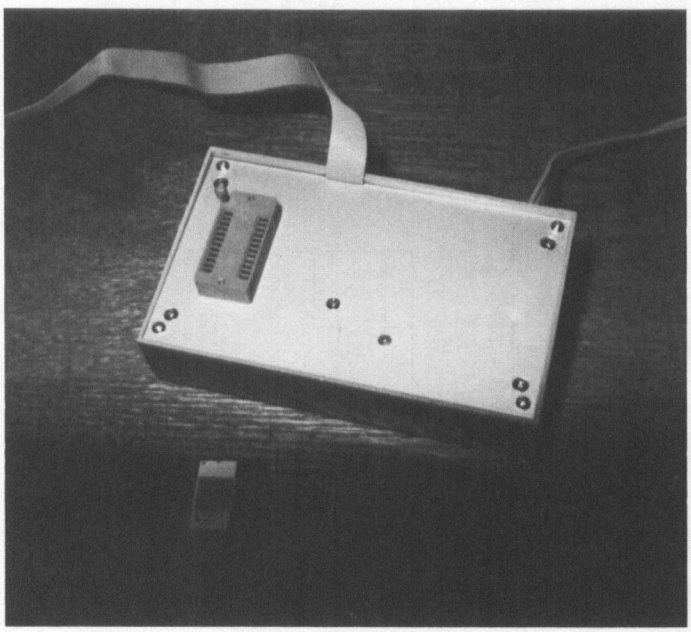
Soll zum Beispiel ein Programm ab einer bestimmten Adresse gespeichert werden, so geht man folgendermaßen vor:



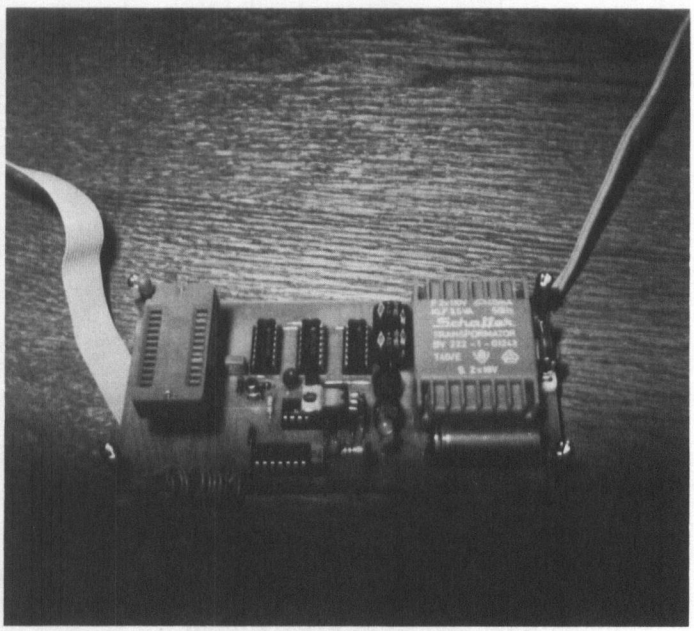

**Schaltung für den CBM-Programmer**

EPROM-Startadresse sei 0100,
? 100/256 = .390625 (=0)
? 100-0 · 256 = 100
POKE 1393,100 : POKE 1394,0
Zur Kontrolle: PRINT PEEK(1393)+256·PEEK(1394) muß wieder 0100 ergeben.
Ebenso muß die Endadresse geändert werden, mit dem Unterschied, daß hier nicht von 000, sondern von 1536 ausgegangen wird.
Es wird allerdings in der Praxis kaum vorkommen, daß ein EPROM teilweise programmiert wird. Es läßt sich bei dieser Methode lediglich die Programmierzeit, die bei 2K immerhin 100 Sekunden beträgt, verringern.
Vor dem Laden des Programms empfiehlt es sich, das Programmiergerät, das ein separates Netzteil enthält, einzuschalten, um Netzstörungen bei nachträglichem Einschalten zu vermeiden.
Bei Programmstart erfolgt zuerst eine Initialisierung des User-Ports. Anschließend erscheint die Bildschirmanzeige:
* EPROGRAMMER 2716 *
MODE-INPUT:
1 READ
2 WRITE
Nach Eingabe des Modus (es ist nur 1 oder 2 möglich) wird z.B. bei Input = 1 der Inhalt eines vorhanden ROM in den RAM-Speicher eingelesen und anschließend automatisch verifiziert, um eine fehlerhafte Datenübernahme auszuschließen. Wurde ein Datenfehler entdeckt, so erscheint auf dem Bildschirm: LOAD ERROR IN XXXX.
Bei Input = 2 wird der Inhalt des EPROM auf 1 = FF überprüft, um sicherzustellen, daß die Speicherplätze nicht schon belegt sind. Sollte dies der Fall sein, so gibt der Rechner PROM ERROR IN XXXX und bricht das Programm ab.
Ist das EPROM einwandfrei, so erfolgt der Schreibvorgang, der sich am deutlichen Flackern der Kontrollampe überwachen läßt.
Nach diesem Vorgang erfolgt wieder eine Verifizierung zur Übernahmekontrolle.
Der Nachbau ist unproblematisch, es ist nur darauf zu achten, daß zwischen der Zählerrücksetzleitung (EOI Pin 4) und den übrigen Anschlüssen im Flachbandkabel eine Masseleitung angeschlossen wird, um Fehlimpulse durch Übersprechen zu vermeiden. Die Kontrolle der Monoflopzeit ist durch die LED möglich, die deutlich sichtbar flackern muß. Mit einem Oszilloskop ist eine exakte Überprüfung möglich, jedoch ist die Impulsdauer unkritisch. Eine Programmierung erfolgt selbst noch bei 40 ms (typ. 45 ms).
Während des Programmierens sollte es vermieden werden, in der Nähe des Rechners befindliche Geräte ein- oder auszuschalten.

**CBM-Computer mit dem Programmer**



**CBM-Programmer mit dem Spannsockel**



**Geöffnetes Gehäuse des CBM-Programmers**

MASCHINENPROGRAMM FUER EPROGRAMMER 2716

```
.: 1024 $0400 00 19 04 01 00 97 30 34 30 2C 30 30 31 3A 97 30
.: 1040 $0410 34 31 2C 30 31 34 3A 8A 00 00 00 00 A9 38 78 8D
.: 1056 $0420 11 E8 A9 8F 8D 10 E8 A9 3C 8D 11 E8 58 A9 89 8D
.: 1072 $0430 10 E8 AD 4C E8 09 C0 29 DF 8D 4C E8 AD 40 E8 09
.: 1088 $0440 08 8D 40 E8 A0 00 B9 97 05 20 D2 FF C8 C0 32 D0
.: 1104 $0450 F5 A9 00 85 A7 C9 01 20 E4 FF F0 F5 E9 30 F0 F1
.: 1120 $0460 AA CA F0 05 CA D0 EA A9 03 A0 01 84 A7 85 FF A9
.: 1136 $0470 F7 2D 11 E8 8D 11 E8 A0 00 84 88 84 89 A5 88 CD
.: 1152 $0480 71 05 D0 07 A5 89 CD 72 05 F0 06 20 19 05 4C 7D
.: 1168 $0490 04 AD 73 05 85 88 AD 74 05 85 89 A6 FF D0 2C CA
.: 1184 $04A0 8E 43 E8 B1 88 8D 4F E8 AD 4C E8 09 20 78 8D 4C
.: 1200 $04B0 E8 A2 F0 A0 27 CA D0 FD 88 D0 FA 29 DF 8D 4C E8
.: 1216 $04C0 58 20 19 05 B0 DD 8C 43 E8 90 30 A9 F7 2D 40 E8
.: 1232 $04D0 8D 40 E8 A6 FF CA F0 11 B1 88 CA F0 02 A9 FF CD
.: 1248 $04E0 4F E8 F0 0A A0 01 88 F0 4F AD 4F E8 91 88 A9 08
.: 1264 $04F0 0D 40 E8 8D 40 E8 20 19 05 B0 D0 A9 08 0D 11 E8
.: 1280 $0500 8D 11 E8 A6 FF F0 08 CA F0 05 CA CA F0 06 60 A9
.: 1296 $0510 02 4C 69 04 A9 00 4C 69 04 E6 88 D0 02 E6 89 A9
.: 1312 $0520 09 8D 10 E8 A9 89 8D 10 E8 AD 76 05 C5 89 90 07
.: 1328 $0530 D0 05 AD 75 05 C5 88 60 A9 08 0D 40 E8 8D 40 E8
.: 1344 $0540 A9 08 0D 11 E8 8D 11 E8 20 D0 FD A6 FF CA CA F0
.: 1360 $0550 05 B9 77 05 D0 03 B9 87 05 20 D2 FF C8 C0 10 D0
.: 1376 $0560 EA A5 89 E9 06 20 75 E7 A5 88 20 75 E7 20 D0 FD
.: 1392 $0570 60 00 00 00 06 FF 0D 12 50 52 4F 4D 20 45 52 52
.: 1408 $0580 4F 52 20 49 4E 20 24 12 4C 4F 41 44 20 45 52 52
.: 1424 $0590 4F 52 20 49 4E 20 24 93 0D 2A 45 50 52 4F 47 52
.: 1440 $05A0 41 4D 4D 45 52 20 32 37 31 36 2A 0D 0D 4D 4F 44
.: 1456 $05B0 45 2D 49 4E 50 55 54 3A 0D 0D 31 20 52 45 41 44
.: 1472 $05C0 0D 32 20 57 52 49 54 45 0D AA AA AA AA AA AA AA
.: 1488 $05D0 AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
.: 1504 $05E0 AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
.: 1520 $05F0 AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
.: 1536 $0600 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
```

RAM SPEICHERBEREICH 2048 BYTE

```
.: 3568 $0DF0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
.: 3584 $0E00 00 0B 0E 0A 00 9E 31 30 35 32 00 00 00
```

Helmut Grothe

# Erweiterung der Peripherie des Tandy TRS-80 Pocket-Computers

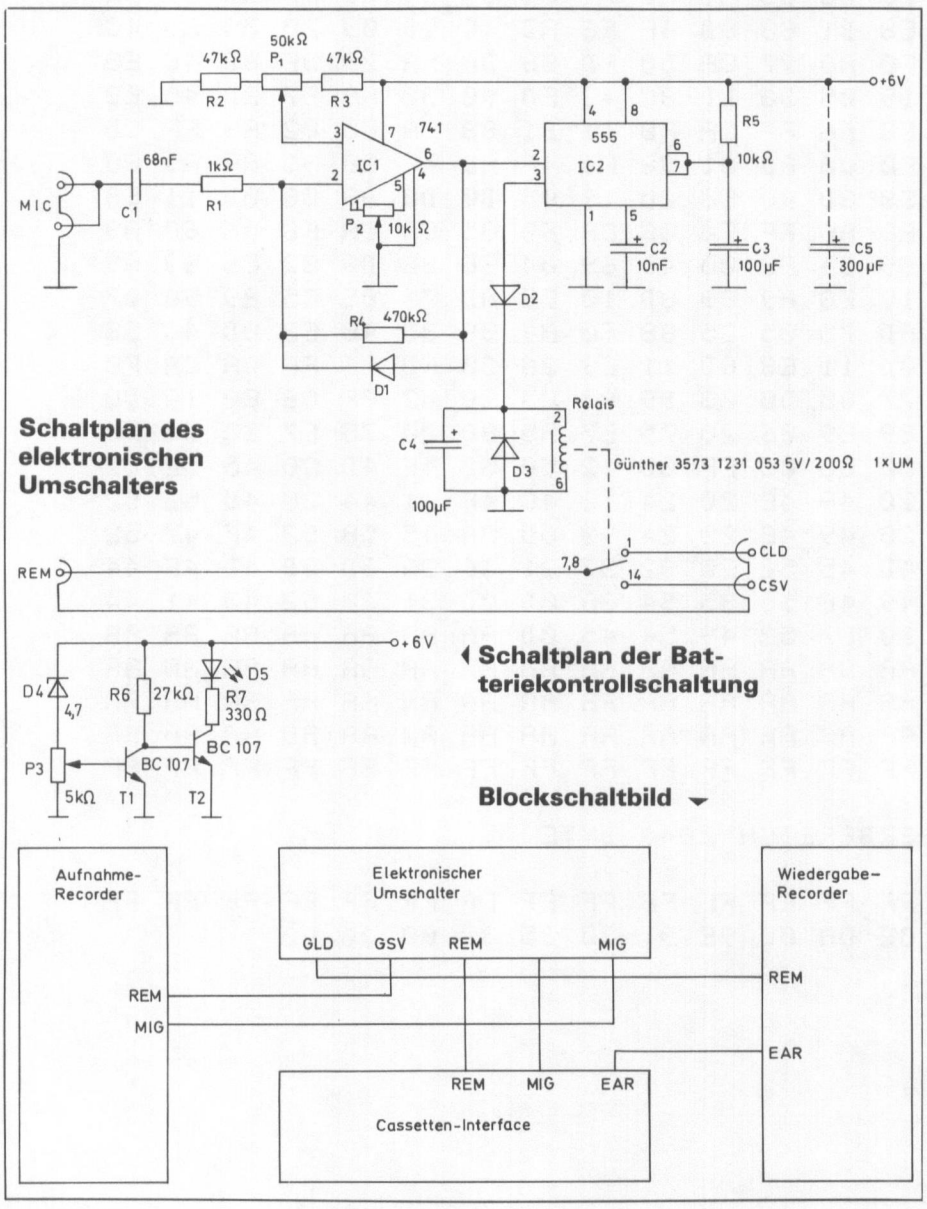
Verfügt man wie beim Tandy TRS-80 Pocket-Computer (identisch mit Sharp PC 1211) über die Möglichkeit, mit den Befehlen CHAIN und INPUT# programmgesteuert Folgeprogramme und Daten vom Kassettenrecorder in den Rechner zu laden und berechnete Datensätze mit dem PRINT#-Befehl dort wieder abzuspeichern, so ist damit trotz der begrenzten internen Speicherkapazität des Pocket-Computers die Bearbeitung umfangreicher Programme und großer Datensätze prinzipiell realisierbar. Die unter Umständen dabei recht häufig auftretende Notwendigkeit, das Kassettengerät manuell von Aufnahme- und Wiedergabebetrieb umzuschalten, erfordert allerdings die ständige Anwesenheit und ungeteilte Aufmerksamkeit des Operateurs, um den vollständigen und reibungslosen Ablauf eines in mehrere Abschnitte unterteilten Programmes mit wiederholter Dateneingabe und Ausgabe zu gewährleisten.

Mit geringem Aufwand läßt sich hier Abhilfe schaffen. Der folgende Beitrag beschreibt Aufbau, Abgleich und Verwendung einer leicht nachzubauenden Schaltung, die – ohne Eingriffe in den möglicherweise noch unter Garantie stehenden Rechner oder sein Kassetteninterface – mit den am Interfaceausgang verfügbaren Signalen die Steuerung zweier Kassettenrecorder übernimmt. Beim Abspeichern von Programmen oder Daten auf Kassette wird automatisch der Motor des auf Aufnahme geschalteten Gerätes in Betrieb genommen. Sollen hingegen von der Kassette Folgeprogramme oder Daten in den Computer geladen werden, so wird für die Dauer dieser Übertragung der Motor des Wiedergabegerätes eingeschaltet.

Am Kassetteninterface des Pocket-Computers sind drei Ausgänge vorhanden: Die REM-Buchse wird während einer Datenübertragung im Interface kurzgeschlossen und setzt damit den Motor des Recorders in Betrieb. An der MIC-Buchse werden die auf Kassette abzuspeichernden Signale ausgegeben, der EAR-Anschluß dient als Eingang für vom Recorder ins Interface laufende Informationen.

Die Aufgabe der gezeigten Schaltung besteht darin, beim Auftreten von Signalen an der MIC-Buchse den REM-Kontakt vom Wiedergabe- auf den Aufnahmerecorder umzuschalten. Liegen an der MIC-Buchse keine Signale an, so bewirkt das Schließen des REM-Kontaktes im Interface die Inbetriebnahme des Wiedergabegerätes.

An der MIC-Buchse des Kassetteninterface tritt vor der eigentlichen Datenübertragung etwa fünf Sekunden lang eine Sinusspannung mit einer Amplitude von wenigen Millivolt auf, die bereits die Umschaltung zwischen den beiden Kassettenrecordern auslöst. Dazu wird die Wechselspannung mit IC1 etwa 500fach verstärkt, wobei die Diode D1 die positiven Halbwellen unterdrückt. Die Gleichspannung am Ausgang von IC1 wird mit dem Trimmer P1 auf etwas mehr als ein Drittel der Versorgungsspannung eingestellt. Sinkt nun durch die negative Halbwelle der überlagerten Wechselspannung der Momentanwert der Ausgangsspannung von IC1 unter ein Drittel der Versorgungsspannung ab, so wird IC2 getriggert. Am Ausgang dieses als Monoflop




**Schaltplan des elektronischen Umschalters**

**Schaltplan der Batteriekontrollschaltung**

**Blockschaltbild**

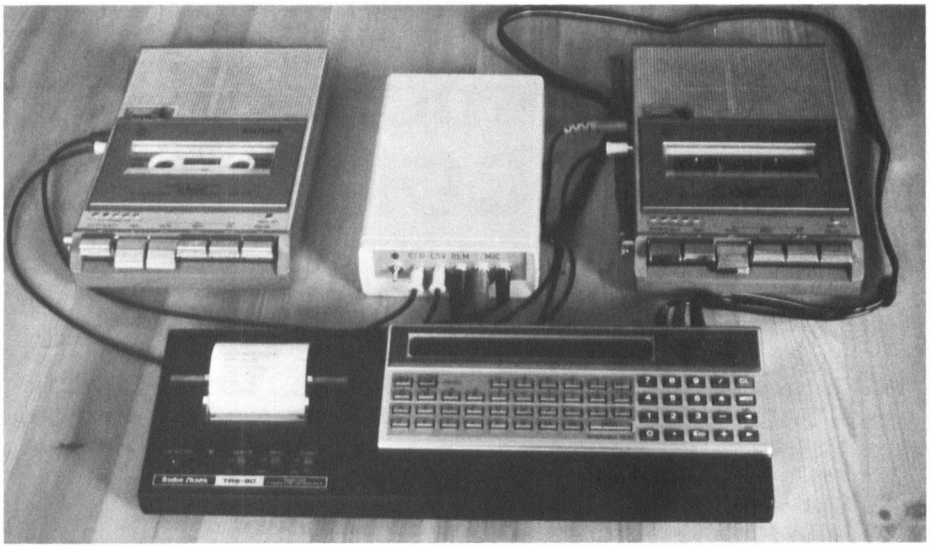
beschalteten Timers wird daraufhin ein etwa eine Sekunde langer Impuls erzeugt, das Relais zieht an und schaltet den REM-Kontakt auf den Aufnahmerecorder um. Solange die Datenübertragung andauert, wird der Monoflop nach jedem Sekundenimpuls sofort wieder neu getriggert. Die Diode D2 und der Kondensator C4 verhindern, daß das Relais während der kurzen Impulspausen abfällt. Die Diode D3 vermeidet das Auftreten von Spannungsspitzen beim Abschalten des Relais. Diese Diode ist in das verwendete Reed-Relais bereits vom Hersteller mit eingebaut.

Die Spannungsversorgung der Schaltung kann wegen des geringen Ruhestrombedarfs von etwa 6 mA aus Batterien erfolgen. Bei Verwendung eines Netzgerätes ist eventuell noch der Kondensator C5 zur Unterdrückung von Störungen einzufügen. Zur Kontrolle des Ladezustands der vier 1,5-V-Batterien dient die Spannungsüberwachungsschaltung. Die LED D5 leuchtet – ähnlich wie beim Kassetteninterface des Pocket-Computers – erst auf, wenn die Versorgungsspannung auf den mit dem Trimmpotentiometer P 3 eingestellten Wert abgesunken ist.

Für den Aufbau des Mustergerätes wurde ein Gehäuse mit eingebauter Batteriehalterung für vier Mignonzellen verwendet. Die IC und das in einem IC-Gehäuse untergebrachte Relais sind auf Stecksockeln montiert.

Auf der Frontplatte des Gerätes sind die beiden MIC-Buchsen parallelgeschaltet (3,5 mm Klinkenbuchsen). Die Verbindung zur Platine sollte möglichst kurz sein, eventuell ist ein abgeschirmtes Kabel zu verwenden. Die Außenleiter der REM-, CSV- und CLD-Buchsen (2,5 mm Klinkenbuchse) sind direkt miteinander verbunden. Der Innenleiter der REM-Buchse wird auf den Umschaltkontakt des Relais gelegt, der Innenleiter der CLD-Buchse ist mit dem Ruhekontakt, der Innenleiter der CSV-Buchse mit dem Arbeitskontakt des Relais verbunden. Über dem Einschalter ist die Batteriekontroll-LED angeordnet. Die Frontplatte besteht aus Kunststoff, um Potentialtrennung zwischen den Buchsen zu gewährleisten. Bei einer Metallfrontplatte müssen die beiden MIC-Buchsen eventuell isoliert montiert werden, um in Verbindung mit dem Recorder Masseschleifen und damit Brummstörungen zu vermeiden.

Die Verschaltung des automatischen Umschalters mit dem Interface und den beiden Kassettenrecordern ist einfach. Es werden dazu noch drei zusätzliche Verbindungskabel benötigt, eines mit 3,5 mm Klinkensteckern und zwei mit 2,5 mm Klinkensteckern an jedem Ende.



**Gesamtansicht der TRS-80-Anlage mit Erweiterung**

Die MIC-Buchse des Interface wird mit der MIC-Buchse des Umschalters verbunden, die zweite parallelgeschaltete MIC-Buchse des Umschalters mit der MIC-Buchse des in Aufnahmestellung befindlichen Recorders. Die EAR-Buchse des Interface wird an die EAR-Buchse des Wiedergaberecorders angeschlossen. Die REM-Buchsen von Interface und Umschalter werden ebenfalls zusammengeschaltet. Von der CLD-Buchse des Umschalters wird ein Kabel zur REM-Buchse des Wiedergaberecorders geführt. Damit wird beim CLOAD-, INPUT#- oder CHAIN-Befehl dieser Recorder angesteuert. Die CSV-Buchse wird mit der REM-Buchse des Aufnahmerecorders verkabelt. Bei CSAVE und PRINT# wird nun dieser Recorder in Betrieb gesetzt.

Der CLOAD?-Befehl kann bei Benutzung des automatischen Umschalters zwar ebenfalls verwendet werden, hier ist jetzt aber nicht nur das Zurückspulen der Kassette notwendig, sondern auch noch deren Umsetzen vom Aufnahme- in den Wiedergaberecorder. Der automatische Umschalter ist eben vor allem für die programmgesteuerten Befehle CHAIN, PRINT# und INPUT# ausgelegt, um manuelle Eingriffe während des Programmablaufs überflüssig zu machen.

Der Abgleich des Umschalters kann notfalls auch ohne Meßgeräte erfolgen. Dazu wird zunächst ein noch ungebrauchter Batteriesatz (Spannung maximal 6,7 V) eingelegt und die Verbindung mit dem Interface und mit den Recordern hergestellt. Die Trimmer P1 und P2 werden in Mittelstellung gebracht. Beim Einschalten der Versorgungsspannung kann das Relais eine Sekunde lang ansprechen (diese Einschaltstörung wird durch C5 ebenfalls behoben) und muß dann wieder abfallen. Sollte dies nicht der Fall sein, so ist P1 langsam im Gegenuhrzeigersinn zu drehen (die Abfallverzögerung beträgt eine Sekunde). Jetzt wird dem Computer ein Ausgabebefehl erteilt (z.B. CSAVE „A"). Ertönt das Vorlaufsignal für die Datenübertragung, so muß das Relais sofort anziehen und den Aufnahmerecorder starten (andernfalls muß P1 im Uhrzeigersinn gedreht werden). Ist die Datenübertragung beendet, muß das Relais spätestens nach einer Sekunde wieder abfallen (sonst P1 im Gegenuhrzeigersinn drehen). Ein eventuell nötiger Feinabgleich kann mit P2 erfolgen. Mit dem bei 6,7 V Batteriespannung vorgenommenen Abgleich arbeitete das Mustergerät störungsfrei bis zu einer Versorgungsspannung von 5,3 V. Die Batteriekontrolle wurde mit P3 so eingestellt, daß die LED bei 5,5 V aufleuchtet.

Die Schaltung kann im Prinzip auch mit höherer Spannung (maximal 18 V) betrieben werden. Dazu müssen die Z-Diode D4, der LED-Vorwiderstand R7 und die Spulenspannung des Relais der geänderten Versorgungsspannung angepaßt werden. Außerdem wird die Schaltung störanfälliger, da mit wachsender Batteriespannung auch die Triggerspannung von IC2 anwächst, die Amplitude der schaltenden Wechselspannung aber praktisch nicht beeinflußt wird. Eventuell muß daher die Verstärkung von IC1 erhöht werden.

Außer für den Tandy TRS-80 Pocket-Computer kann der beschriebene Umschalter natürlich auch bei allen anderen Computern eingesetzt werden, die über programmgesteuerte Ein- und Ausgabe mit Kassettenrecordern verfügen.

Wolfgang Oertwig

# Ein Experimentierbus für EUROCOM 1

Sehr viele Hobbyelektroniker stehen vor der Notwendigkeit sich mit der Technik der Mikroprozessoren vertraut zu machen. Hier bieten sich die Einplatinen-Mikrocomputer geradezu an, da bei diesen Geräten die Elektronik noch „hautnah" erlebt werden kann. Doch die fast unüberschaubare Vielfalt von verschiedenen Typen und Anbietern erschwert die Entscheidung erheblich. Zumal eine Einsicht in das mitgelieferte Begleitmaterial vor dem Kauf normalerweise nicht möglich ist.

Doch auch wenn die Wahl für den Anfang richtig und zufriedenstellend getroffen wurde, kommt sehr schnell der Punkt, wo man an die Grenzen des Systems stößt. Viele Firmen bieten für ihre Systeme neben dem Grundausbau noch die unterschiedlichsten Erweiterungen an. Aber kann der Hobbyelektroniker seine eigenen Vorstellungen und Wünsche einer Systemerweiterung realisieren? Lassen sich bereits vorhandene Platinen mit einer anderen Steckbelegung anschließen?

Diese Lücke soll jetzt mit dem Experimentierbus, der Bus-Platine zum Erweitern und Experimentieren, geschlossen werden.

## Das Grundkonzept

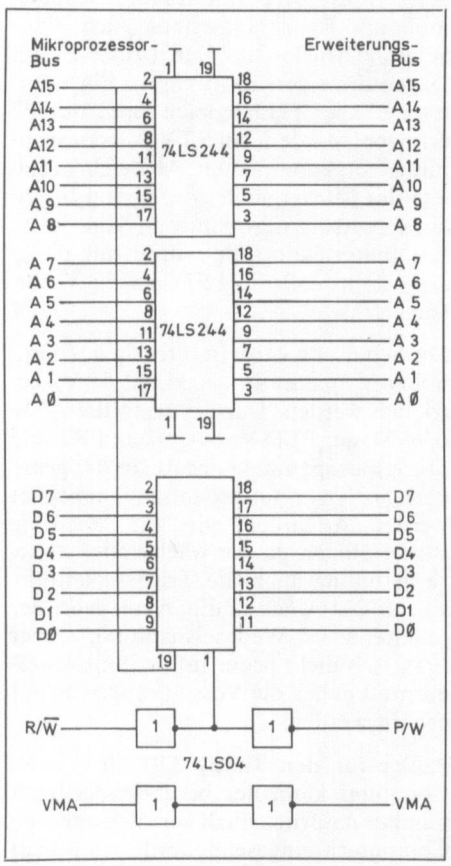
Sehr viele Einplatinencomputer sind als Europakarte oder als Doppeleuropakarte ausgelegt und benutzen als Steckverbinder eine 64polige Messerleiste nach DIN 41612 Bauform C Reihe a+c. Diese Steckverbinder sind auch als VG-Leiste bekannt.

Diesen Gegebenheiten sollte sich der Experimentierbus anpassen,und so entstand eine Europakarte mit einem Rangierfeld und einem Erweiterungsbus mit drei Steckplätzen.

## Das Prinzip des Rangierfeldes

Das Rangierfeld erlaubt es prinzipiell jeden Punkt des Eingangsbusses mit jedem Punkt des Erweiterungsbusses zu verbinden.

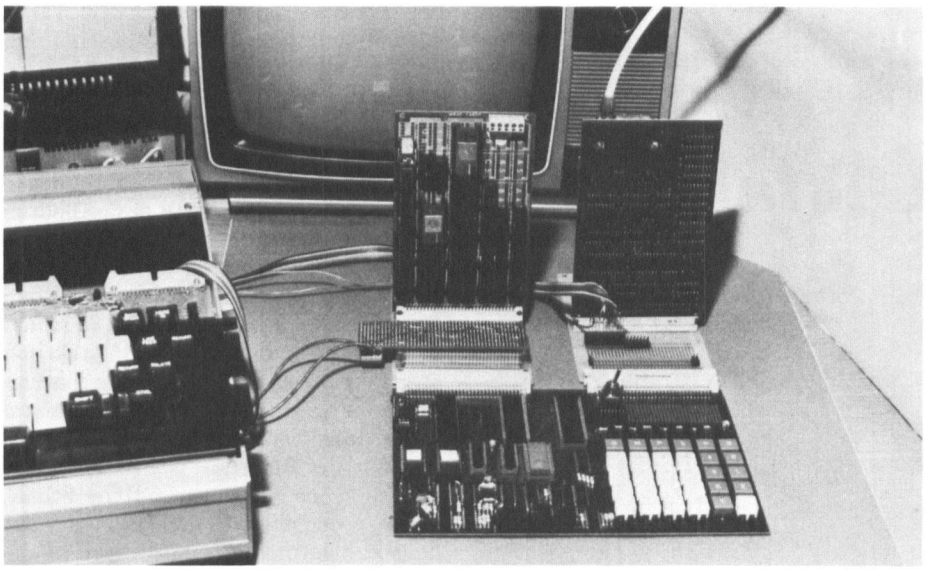
Das Rangierfeld besteht aus zwei Stekkerplätzen, die beliebig bestückt werden können. Die Signale des Eingangsbusses Pinreihe a sind im Rangierfeld auf den Stecker R1 Pinreihe a gelegt. Vom Stek-



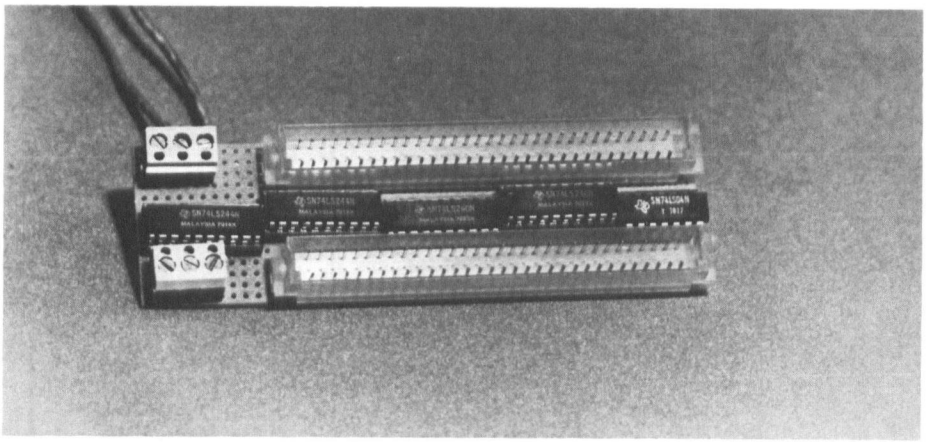

**Experimentierbus-Platten zum Anschluß des Videointerfaces VIC 1**

### Rangiertabelle
### Rangierfeldbelegung und Zuordnung

| Bus-Pin Eing. + Ausg. Bus Reihe a | Rangierfeld Eingang R1 Reihe a | Ausgang 0° R1 Reihe c | Ausgang 180° R2 Reihe c | Bus Pin Ein- + Ausg. Bus Reihe c | Rangierfeld Eingang R2 Reihe a | Ausgang 0° R2 Reihe c | Ausgang 180° R1 Reihe c |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 32 | 1 | 1 | 1 | 32 |
| 2 | 2 | 2 | 31 | 2 | 2 | 2 | 31 |
| 3 | 3 | 3 | 30 | 3 | 3 | 3 | 30 |
| 4 | 4 | 4 | 29 | 4 | 4 | 4 | 29 |
| 5 | 5 | 5 | 28 | 5 | 5 | 5 | 28 |
| 6 | 6 | 6 | 27 | 6 | 6 | 6 | 27 |
| 7 | 7 | 7 | 26 | 7 | 7 | 7 | 26 |
| 8 | 8 | 8 | 25 | 8 | 8 | 8 | 25 |
| 9 | 9 | 9 | 24 | 9 | 9 | 9 | 24 |
| 10 | 10 | 10 | 23 | 10 | 10 | 10 | 23 |
| 11 | 11 | 11 | 22 | 11 | 11 | 11 | 22 |
| 12 | 12 | 12 | 21 | 12 | 12 | 12 | 21 |
| 13 | 13 | 13 | 20 | 13 | 13 | 13 | 20 |
| 14 | 14 | 14 | 19 | 14 | 14 | 14 | 19 |
| 15 | 15 | 15 | 18 | 15 | 15 | 15 | 18 |
| 16 | 16 | 16 | 17 | 16 | 16 | 16 | 17 |
| 17 | 17 | 17 | 16 | 17 | 17 | 17 | 16 |
| 18 | 18 | 18 | 15 | 18 | 18 | 18 | 15 |
| 19 | 19 | 19 | 14 | 19 | 19 | 19 | 14 |
| 20 | 20 | 20 | 13 | 20 | 20 | 20 | 13 |
| 21 | 21 | 21 | 12 | 21 | 21 | 21 | 12 |
| 22 | 22 | 22 | 11 | 22 | 22 | 22 | 11 |
| 23 | 23 | 23 | 10 | 23 | 23 | 23 | 10 |
| 24 | 24 | 24 | 9 | 24 | 24 | 24 | 9 |
| 25 | 25 | 25 | 8 | 25 | 25 | 25 | 8 |
| 26 | 26 | 26 | 7 | 26 | 26 | 26 | 7 |
| 27 | 27 | 27 | 6 | 27 | 27 | 27 | 6 |
| 28 | 28 | 28 | 5 | 28 | 28 | 28 | 5 |
| 29 | 29 | 29 | 4 | 29 | 29 | 29 | 4 |
| 30 | 30 | 30 | 3 | 30 | 30 | 30 | 3 |
| 31 | 31 | 31 | 2 | 31 | 31 | 31 | 2 |
| 32 | 32 | 32 | 1 | 32 | 32 | 32 | 1 |



**EUROCOM 1 Mit Videointerface und Erweiterungskarte mit der Steckerbelegung für ein anderes Mikrocomputersystem**



**Treiberstufe für Adressen- und Datenleitungen**

### Anschlußbelegung der VG-Leisten

| EUROCOM I a | EUROCOM I c | Pin-Nr. | System X a | System X c |
|---|---|---|---|---|
| GND | GND | 1 | +5 V | +5 V |
| CB2 | A15 | 2 | $\overline{\text{IRQ}}$ | – – |
| CB1 | A14 | 3 | $\overline{\text{NMI}}$ | RE-SET |
| PB7 | A13 | 4 | VMA | R/W |
| PB6 | A12 | 5 | GND | GND |
| PB5 | A11 | 6 | Ø2 | Ø1 |
| PB4 | A10 | 7 | GND | GND |
| PB3 | A9 | 8 | MEMC LK | – – |
| PB2 | A8 | 9 | – – | – – |
| PB1 | A7 | 10 | – – | – – |
| PBØ | A6 | 11 | – – | – – |
| PA7 | A5 | 12 | – – | – – |
| PA6 | A4 | 13 | – – | – – |
| PA5 | A3 | 14 | – – | – – |
| PA4 | A2 | 15 | +12 V | +12V |
| PA3 | A1 | 16 | – – | – – |
| PA2 | AØ | 17 | GND | GND |
| PA1 | R/W | 18 | – – | – – |
| PAØ | DØ | 19 | – – | – – |
| $\overline{\text{IRQ}}$ | D1 | 20 | $\overline{\text{D3}}$ | $\overline{\text{D1}}$ |
| CA2 | D2 | 21 | $\overline{\text{D7}}$ | $\overline{\text{D5}}$ |
| CA1 | D3 | 22 | $\overline{\text{D2}}$ | $\overline{\text{DØ}}$ |
| $\overline{\text{NMI}}$ | D4 | 23 | $\overline{\text{D6}}$ | $\overline{\text{D4}}$ |
| $\overline{\text{HALT}}$ | D5 | 24 | A15 | A14 |
| BA | D6 | 25 | A13 | A12 |
| RXD | D7 | 26 | A11 | A10 |
| TXD | $\overline{\text{RES}}$ | 27 | A9 | A8 |
| VMA | E | 28 | A7 | A6 |
| MR | 300B | 29 | A5 | A4 |
| +12 V | −5 V | 30 | A3 | A2 |
| +5 V | + 5V | 31 | A1 | AØ |
| GND | GND | 31 | GND | GND |

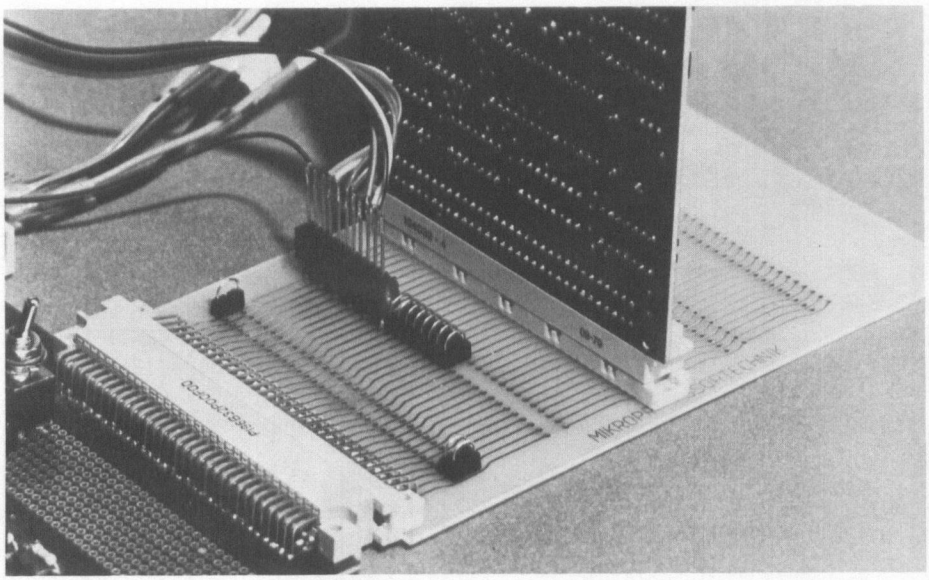
kerplatz R1 Pinreihe c geht die Verbindung weiter zur Pinreihe a des Erweiterungsbusses. Das gleiche Prinzip findet für die Pinreihe c auf dem Steckerplatz R2 Anwendung.
Hier wird die Pinreihe c des Eingangsbusses auf den Stecker R2 Pinreihe a im Rangierfeld gelegt und von der Pinreihe c geht es weiter zur Pinreihe c des Erweiterungsbusses.
Da somit der Eingangs- und der Erweiterungsbus vollständig voneinander getrennt sind, eröffnen sich unzählige Möglichkeiten einer Signalrangierung.

## Die einfache Buserweiterung

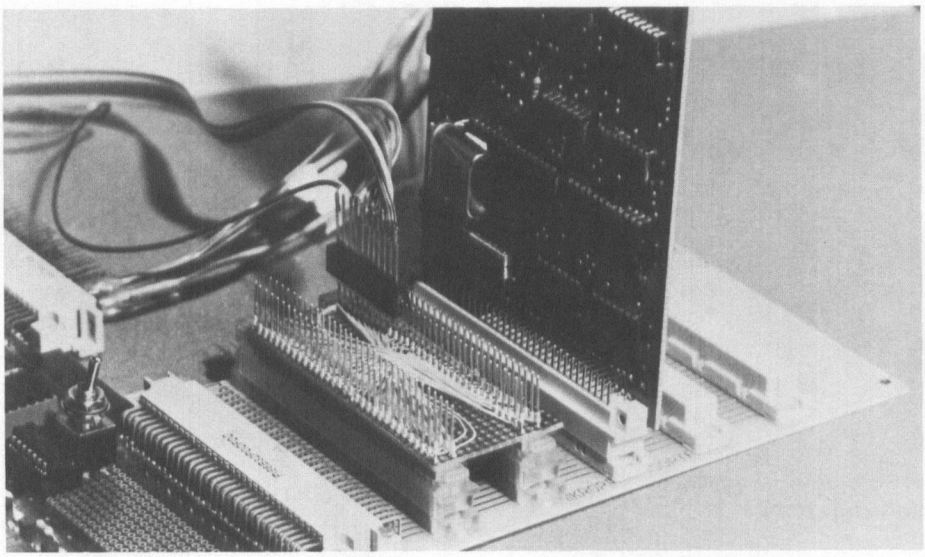
In diesem Anwendungsfall ist das Rangierfeld eigentlich überflüssig. Doch da die Bustrennung nun einmal vorhanden ist, muß sie durch Lötbrücken wieder aufgehoben werden. Im Rangierfeld müssen auf den Steckerplätzen R1 und R2 die Punkte der Pinreihen a mit den Punkten der Pinreihen c verbunden werden. Hierzu eignet sich z.B. der Löt-Brükkenbaustein LBB 126.
Bei dem Aufstecken der Erweiterungskarten zeigt die Lötseite zum Eingangsstecker. Wie bei jedem Bussystem stehen die Karten alle in einer Richtung, nur ist hier die Treiberkarte (der Einplatinencomputer) um 90° zur Seite geklappt. Für einige Anwendungsfälle wäre es aber wünschenswert, wenn die Bauteileseite der Erweiterungskarte zum Eingangsstecker zeigen würde. So z.B. wenn Leuchtdioden oder andere Anzeigeelemente sichtbar sein sollen. Hierzu müßte der Erweiterungsbus um 180° gedreht werden.
Die einfache Buserweiterung mit um 180° gedrehtem Erweiterungsbus ist mit dem Rangierfeld realisierbar. Hier kann man entweder die notwendigen Verbindungsdrähte im Rangierfeld direkt löten oder Wire-Wrap-Stifte einsetzen. Die Verbindungen lassen sich dann in der Wire-Wrap-Technik (Wickeltechnik) jederzeit lösbar realisieren. Welche Punkte des Eingangs- und des Ausgangsbusses im Rangierfeld zu finden sind ist der Tabelle 1 zu entnehmen.

## Anschluß der Videointerface-Platine VIC I an einen EUROCOM 1

Zum komfortablen Dialogbetrieb über den Fernsehbildschirm mit dem EUROCOM 1 muß neben der Videointerface-Platine auch noch eine ASCII-Tastatur vorhanden sein und angeschlossen werden. Auch in diesem Fall kann der Experimentierbus mit Rangierfeld sinnvoll eingesetzt werden.
Die Verbindung für die Stromversorgung der Interfaceplatine und die Steuersignale werden mit den Lötbrückenbausteinen LBB 126 realisiert und der Tastaturanschluß erfolgt über Wire-Wrap-Stif-



**Rangiermodul mit Interfaceelektronik**



**Experimentierbus-Platine mit Rangiermodul und Videointerface**

te, auf die ein IC-Sockel mit den Verbindungsleitungen zur Tastatur aufgesteckt wird. Bei dieser Anwendung wurde der Rangieraufwand und der Ausbau des Erweiterungsbusses auf das notwendige Minimum und somit auf einen speziellen Anwendungsfall beschränkt. Das bedeutet, daß für verschiedene Einsatzfälle immer eine neue Experimentierbus-Platine notwendig wird.

## Universeller Einsatz durch „Rangiermodule"

Um nicht für jeden Einsatzfall eine neue Experimentierbus-Platine zu benötigen, lassen sich auch im Rangierfeld die 64poligen Buchsenleisten einsetzen. Durch Aufstecken von passenden Messerleisten, die über eine kleine Leiterplatte mechanisch gekoppelt sein sollten, entsteht ein Rangiermodul, auf dem die gewünschten Rangierungen vorgenommen werden können. Es wird nur eine Platine benötigt, auf die je nach Anwendungsfall der entsprechende Rangiermodul aufgesteckt wird. Das Bild zeigt einen Rangiermodul mit der notwendigen Verdrahtung für den um 180° gedrehten Erweiterungsbus in Wire-Wrap-Technik. Dieser universelle Rangiermodul kann auch zum Anschluß der Videointerface-Platine VIC 1 an den EUROCOM 1 Verwendung finden.

## Rangiermodul mit Interfaceelektronik

Auf einem solchen Rangiermodul lassen sich aber ebensogut einige integrierte Schaltkreise als Interfaceelektronik unterbringen. Denkbar ist eine Treiberelektronik für die Adreß- und die Datenleitungen, die es erlaubt, Erweiterungskarten mit höheren Lastfaktoren anzuschließen, deren Betrieb sonst wegen der begrenzten Ausgangsleistung der Grundkarte nicht möglich wäre. Die Schaltung zeigt eine solche Treiberstufe.

Hierbei muß aber beachtet werden, daß die Treiber für den Datenbus nicht gegen die vorhandene Elektronik auf der Platine arbeitet. Beim EUROCOM 1 liegen die internen Speicher und Peripheriebausteine alle in dem Adressenbereich, der die Adressenleitung A15 = High benötigt. Mit diesem Signal kann man den Datenbustreiberbaustein 74LS245 in den Tri-State-Zustand bringen, d.h. praktisch abschalten. Damit steht aber für Systemerweiterungen nur noch der Bereich von 0000 (Hexa) bis 7FFF (Hexa) gleich 0 bis 32K zur Verfügung.

Wie zu sehen ist, lassen sich die Schaltkreise zwischen den Steckerleisten auf dem Rangiermodul unterbringen.

So bereitet es auch keine Schwierigkeiten, Karten mit anderen Steckerbelegungen und z.B. invertierten Signalen anzuschließen.

Das Foto zeigt eine angeschlossene Prototypkarte, die aus einem anderen Mikrocomputersystem stammt und eine ganz andere Steckerbelegung hat als der EUROCOM. Tabelle 2 zeigt die beiden Steckerbelegungen im Vergleich.

Wie der Steckerbelegung des Systems X zu entnehmen ist, werden die Datenleitungen mit invertierten Signalpegeln gesteuert. Die Interfaceelektronik muß nun auch die Signalumkehr der Datenleitungen übernehmen. Bei einem geeigneten Aufbau lassen sich durch den Einsatz verschiedener pinkompatibler Treiber-Bausteine sowohl invertierende und auch nicht invertierende Ausgangsleitungen realisieren.

| | nicht invertierend | invertierend |
|---|---|---|
| Datenbus | 74LS244 | 74LS240 |
| Adreßbus | 74LS244 | 74LS240 |

Auch hierbei muß durch eine geeignete Verriegelung verhindert werden, daß die Treiberstufen gegen die vorhandene Elektronik auf der Platine arbeitet.

Durch diese Beispiele wurde versucht, die vielseitigen Anwendungsmöglichkeiten der Experimentierbus-Platine aufzuzeigen.

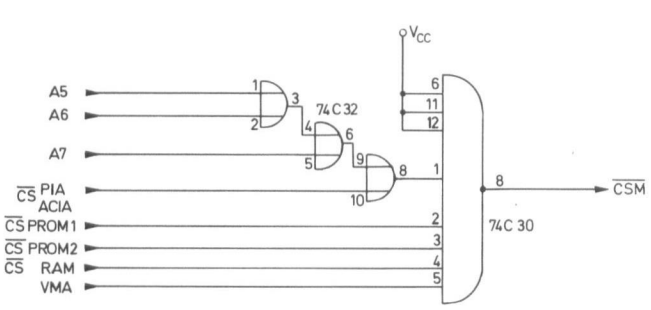
Um den Anwendungsbereich noch weiter zu vergrößern, wurde die Platine noch einmal weiterentwickelt und somit wird auch der Anschluß von Karten möglich, die eine 64polige VG-Leiste mit der Pinreihe a + b haben. Die Pinreihen b und c wurden gebrückt und sind somit potentialmäßig identisch.

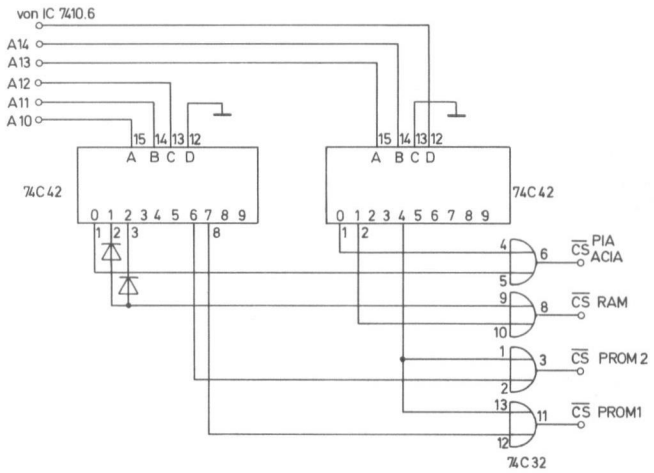
Wilfried Hofmann

# Hardwaretips für EUROCOM 1

Um den vollen Speicherbereich meines „EUROCOM 1" zu nutzen, mußte ich die CS-Signale umändern. Bei der ursprünglichen CS-Auswahl gab es Schwierigkeiten bei der Belegung einiger Adressen. Um hier nur ein Beispiel zu nennen: Wird die Adresse 9000 H angesprochen, so entsteht gleichzeitig bei Adr. 8000 H ein CS. Um dies zu verhindern, mußte ich die Adressenleitungen A12 und A13 ebenfalls dekodieren. Dies erreichte ich durch die Zusatzverdrahtung zweier IC, die auf dem Beschriftungsfeld links neben der Tastatur leicht Platz fanden.

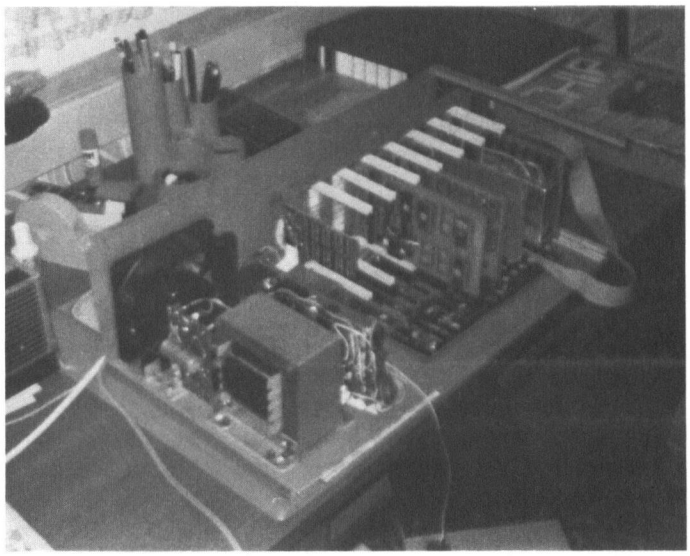
Bei späteren Speichererweiterungen dient der Bereich A400-A77FF nur noch für Stack usw. Hierfür sind auch die beiden Dioden, die diesen Bereich gleichzeitig für Adr. A000-A3FF zugänglich machen. Dies ist der Stack-Bereich des MIK-BUG-Monitors.
Weiterhin mußte ich Bus-Treiber in die Daten- und Adressenleitungen einbauen. Dazu brauchte ich ein Signal, das mir die Treiber-IC durchschaltet, wenn eine Adresse angesprochen wird, die nicht auf der CPU-Platine liegt. Dazu nahm ich die vier CS-Signale, VMA und A5, A6, A7. Diese verknüpfte ich und erhielt so das erwünschte Signal CSM. Liegt eines der Signale CS PROM 1, CS PROM 2 oder CS RAM auf L-Pegel, so ist am Ausgang des NAND-Gatter ein H-Pegel. Das Signal CS PIA wirkt sich nur am Ausgang aus wenn keine der Adressenleitungen A5, A6, A7 high sind. Für diese drei können auch andere Adressen genommen werden, die nicht im Zusammenhang mit CS PIA auf der CPU-Platine gebraucht werden. Sie haben lediglich die Aufgabe, Adressen im Bereich 8000-83FF außerhalb der EUROCOM 1 Platine zu bilden, die z.B. für Ports, A/D-Wandler usw. gebraucht werden.




**Schaltung für die Adressierung**

**Erweiterungsschaltung für den EUROCOM 1**



**Gesamtansicht des Computergerätes, das der Autor auf der Basis des EUROCOM 1 aufgebaut hat**

Peter Weiß

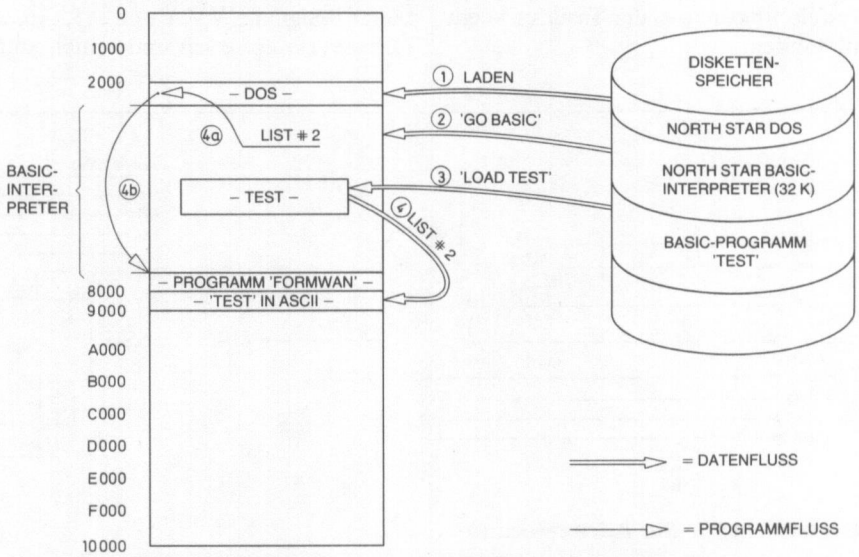
# NORTH-STAR-BASIC im ASCII-Format

Mit „Kompatibilität" kann man ein dunkles Kapitel im Mikro-Computer-Hobby überschreiben. Es kommt immer wieder vor, daß man bei seinen Entwicklungen an eine Grenze stößt, die nur durch neue Anschaffungen zu überwinden ist. In der Regel kann man damit einen Teil dessen, was man sich bis dahin erarbeitet hat, nicht mehr oder nur mit Hilfe größerer Umstellungen weiterverwenden.

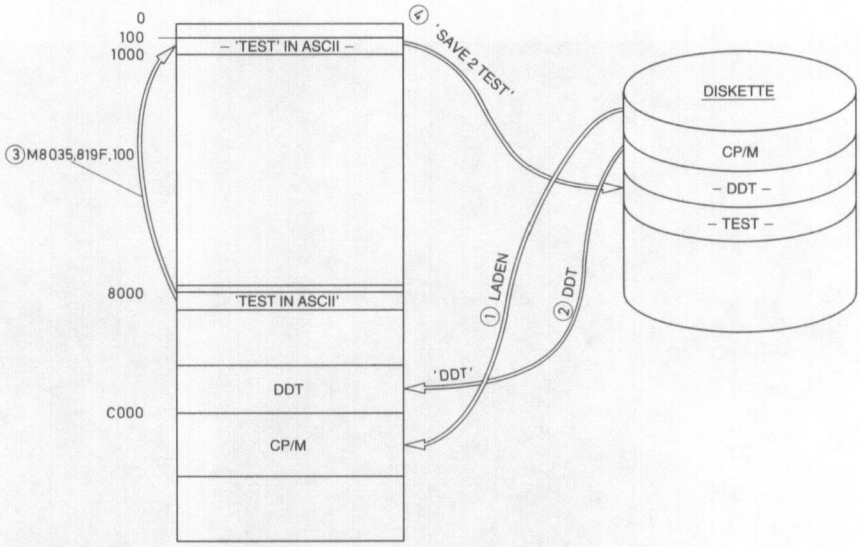
Ein gutes Beispiel dafür ist die Umstellung irgendeines Diskettenbetriebssystemes auf CP/M. Viele Besitzer eines mit Diskettenlaufwerken ausgerüsteten Mikrocomputers, der nicht mit dem weitverbreiteten Programmsystem CP/M geliefert wurde, entschließen sich früher oder später „auf CP/M umzusteigen". Das CP/M bietet dem Benutzer erhebliche Vorteile gegenüber einigen anderen, einfacheren Systemen, von denen als wichtige Vorzüge nur das reiche Programmangebot auf dem Softwaremarkt und die gute Dateiverwaltung genannt werden sollen. Es kann daher sehr sinnvoll sein, dieses Betriebssystem trotz seines relativ hohen Preises zu erwerben.
Besitzt man zum Beispiel einen „North-Star-Horizon"-Computer und hat auch Programme mit dem „North-Star-BASIC" erstellt, so möchte man diese Programme sicherlich auch unter dem neuen Betriebssystem weiterverwenden, ohne sie völlig neu in das System einzugeben.
Nun haben „North-Star-BASIC"-Programme ein eigenes, internes Format, in dem sie auf der Diskette gespeichert werden. Keiner der BASIC-Compiler oder -Interpreter, die unter CP/M arbeiten, könnte dieses Format verarbeiten, abgesehen davon, daß man die Programme nicht einfach von einer „North-Star"-Diskette auf eine CP/M-Diskette kopieren kann, da die Software-Zugriffsmechanismen völlig verschieden sind.
Würde man also ein „North-Star"-BASIC-Programm in den Hauptspeicher laden, dann das CP/M-System in den Hauptspeicher bringen und dann dieses BASIC-Programm in eine CP/M-Datei auf der Diskette sichern, so könnte man mit dieser Datei nichts anfangen. Sie hätte das gezeigte Format.

## Ein kleiner Trick

In diesem Format steht für das BASIC-Wort „PRINT" nur eine hexadezimale Zahl, 82h, für „REM" die Zahl 8fh. Trotzdem ergibt der Befehl „LIST" eine lesbare Liste.
Das „North-Star"-BASIC wandelt auf den LIST-Befehl hin sein internes Format um, zurück in das Standard-ASCII-Format, in dem das Programm eingegeben wurde.
Genau diese Eigenschaft kann man sich zunutze machen, wenn man aus „North-Star-BASIC"-Quellendateien Standard-BASIC-Quellendateien erstellen will.
Man benötigt dazu lediglich ein Pro-




**Laden des NORTH-STAR-BASIC-Programms und Erstellen einer ASCII-Datei im Hauptspeicher**




**Sichern des ASCII-Quellenprogramms auf der CP/M-Diskette**

gramm, das, anstatt die durch den LIST-Befehl erzeugten ASCII-Zeichen zu drucken, diese zu einer Datei sammelt. Die Abbildung zeigt als Beispiel eines solchen Programmes das Programm „FORMWAN“ (FORMAT-WANDLER).

Im folgenden soll erläutert werden, wie mit Hilfe dieses Programmes „North-Star“-BASIC- in ASCII-BASIC-Quellendateien umgewandelt werden können.

Es soll das „North-Star“-BASIC-Programm „TEST“ in ein BASIC-Programm in ASCII-Zeichen umgewandelt werden. Dazu wird zunächst das Programm „FORMWAN“ in den Speicher auf die Adresse 8000 geladen. Dann wird in der „North-Star-DOS“-Bildschirm-/Drukker-Routine (auf der Adresse 2900) der „CALL“-Befehl zur Druckerroutine geändert in einen CALL-Befehl zum Programm „FORMWAN“ (aus „cd d6 29“ wird „cd 00 80“).

Nun lädt man das „North-Star“-BASIC mit dem Befehl „GO BASIC“ und dann das BASIC-Programm „TEST“.

Mit dem Befehl „LIST #2“ würde das Programm „TEST“ auf den Drucker ausgegeben. Da aber logisch anstelle der Druckerroutine das Programm „FORMWAN“ liegt, werden diesem Programm die einzelnen ASCII-Zeichen übergeben und dann als Datei im Speicher abgelegt.

Das Ergebnis dieses „speziellen“ LIST#2-Befehles ist gezeigt. Auf der Adresse 8000H beginnt das Programm FORMWAN. Wie man aus der Programmliste erkennen kann, beginnt der Speicherbereich, in dem die Datei abgelegt wird, auf der Adresse 8035H. Man sieht, daß die Datei mit einem Wagenrücklauf- und einem Neue-Zeile-Zeichen beginnt.

Im nächsten Schritt soll diese Datei, das Programm TEST, in eine CP/M-Datei gesichert werden. Dazu wird zunächst das Betriebssystem CP/M in den Hauptspeicher geladen. Danach ruft man mit dem Befehl „DDT“ das CP/M-Monitorprogramm auf. Mit seiner Hilfe kann die Datei „TEST“ im Speicher an eine Stelle geschoben werden, von der aus es sich leicht auf die Diskette bringen läßt. Dies geschieht mit dem Befehl m, x, y, z, wobei x die erste Adresse des zu übertragenden Speicherbereiches, y die letzte Adresse und z die Zieladresse angibt. Der Wert für x steht fest: Es ist die symbolische Adresse SPBER in der Programmliste, die echte Adresse 8035H. Den Wert für y erhält man, wenn man von der Adresse auf der symbolischen Adresse ZEIGER in der Liste (Adresse 8033) eine 1 subtrahiert. Diese Stelle findet man im Speicherauszug leicht wieder, da sie zwei

**Das zu konvertierende BASIC-Programm „TEST“**

```
100 REM ****************************************
110 REM *   TEST PROGRAMM IN 'NORTH STAR' - BASIC   *
130 REM
140 PRINT \ PRINT " *  QUADRATZAHLEN 1-9:  *" \ PRINT
150 FOR I = 1 TO 9
160 PRINT "      ",I," *",I," =",I*I
170 NEXT I
180 PRINT \ PRINT " **   PROGRAMM ENDE   **" \ PRINT
190 END
200 REM ****************************************
READY
```

**Ausdruck des Programms „TEST“**

```
4000  34 64 00 20  8F 20 E2 E2   E2 E2 E2 E2  E2 E2 E2 E2
       4  d        -     -b -b   -b -b -b -b  -b -b -b -b
4010  E2 E2 E2 E2  E2 E2 E2 E2   E2 E2 E2 E2  E2 E2 E2 E2
      -b -b -b -b  -b -b -b -b   -b -b -b -b  -b -b -b -b
4020  E2 E2 E2 E2  E2 E2 E2 E2   E2 E2 E2 E2  E2 E2 E2 E2
      -b -b -b -b  -b -b -b -b   -b -b -b -b  -b -b -b -b
4030  E2 E2 E2 0D  33 6E 00 20   8F 20 E2 20  20 20 54 45
      -b -b -b     3  n        -     -b          T  E
4040  53 54 20 50  52 4F 47 52   41 4D 4D 20  49 4E 20 27
       S  T     P   R  O  G  R    A  M  M     I  N     '
4050  4E ED 54 48  20 53 54 41   52 27 20 E5  20 42 41 53
       N -m  T  H      S  T  A    R  '    -e     B  A  S
4060  49 43 20 20  20 E2 0D 06   82 00 20 8F  0D 2A 8C 00
       I  C           -b        -        -     * -
4070  20 82 20 5C  20 82 20 22   20 2A 20 20  51 55 41 44
      -     \     -     "        *           Q  U  A  D
4080  52 41 54 5A  41 48 4C 45   4E 20 31 2D  39 3A 20 20
       R  A  T  Z   A  H  L  E    N     1  -   9  :
4090  2A 22 20 5C  20 82 0D 10   96 00 20 81  20 49 20 F5
       *  "     \     -        -        -     I     -u
40A0  20 31 20 B1  20 39 0D 20   A0 00 20 82  20 22 20 20
         1    -1     9          -        -     "
40B0  20 20 20 22  2C 49 2C 22   20 2A 22 2C  49 2C 22 20
            "     ,  I  ,  "        *  "  ,   I  ,  "
40C0  3D 22 2C 49  E2 49 0D 08   AA 00 20 83  20 49 0D 29
       =  "  ,  I  -b  I        -*        -     I     )
40D0  B4 00 20 82  20 5C 20 82   20 22 20 2A  2A 20 20 20
      -4     -     \     -        "     *   *
40E0  50 52 4F 47  52 41 4D 4D   20 45 4E 44  45 20 20 20
       P  R  O  G   R  A  M  M     E  N  D   E
40F0  2A 2A 22 20  5C 20 82 0D   06 BE 00 20  8D 0D 34 C8
       *  *  "     \     -        -)        -     4 -H
4100  00 20 8F 20  E2 E2 E2 E2   E2 E2 E2 E2  E2 E2 E2 E2
         -     -b -b -b -b   -b -b -b -b  -b -b -b -b
4110  E2 E2 E2 E2  E2 E2 E2 E2   E2 E2 E2 E2  E2 E2 E2 E2
      -b -b -b -b  -b -b -b -b   -b -b -b -b  -b -b -b -b
4120  E2 E2 E2 E2  E2 E2 E2 E2   E2 E2 E2 E2  E2 E2 E2 E2
      -b -b -b -b  -b -b -b -b   -b -b -b -b  -b -b -b -b
4130  E2 0D 01 00  00 00 00 00   00 00 00 00  00 00 00 00
      -b
```

**Hauptspeicherauszug, das BASIC-Programm „TEST“ im NORTH-STAR-Format (komprimiert)**

```
8000 E5 3A 2A 80 B7 C2 27 80 2A 33 80 70 23 22 33 80 .:*...'.*3.p#"3.
8010 2A 31 80 23 22 31 80 2A 2B 80 2B 22 2B 80 7C B5 *1.#"1.*+.+"+.|.
8020 20 05 3E FF 32 2A 80 78 E1 C9 00 15 3E DD DD DD  .>.2*.x....>...
8030 DD 6B 01 A0 81 0D 0A 31 30 30 20 52 45 4D 20 2A .k.....100 REM *
8040 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A **********
8050 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A **********
8060 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 0D 0A 31 31 ************..11
8070 30 20 52 45 4D 20 2A 20 20 20 54 45 53 54 20 50 0 REM *   TEST P
8080 52 4F 47 52 41 4D 4D 20 49 4E 20 27 4E 4F 52 54 ROGRAMM IN 'NORT
8090 48 20 53 54 41 52 27 20 2D 20 42 41 53 49 43 20 H STAR' - BASIC
80A0 20 20 2A 0D 0A 31 33 30 20 52 45 4D 0D 0A 31 34   *..130 REM..14
80B0 30 20 50 52 49 4E 54 20 5C 20 50 52 49 4E 54 20 0 PRINT \ PRINT
80C0 22 20 2A 20 20 51 55 41 44 52 41 54 5A 41 48 4C " *  QUADRATZAHL
80D0 45 4E 20 31 2D 39 3A 20 20 2A 22 20 5C 20 50 52 EN 1-9:  *" \ PR
80E0 49 4E 54 0D 0A 31 35 30 20 46 4F 52 20 49 20 3D INT..150 FOR I =
80F0 20 31 20 54 4F 20 39 0D 0A 31 36 30 20 50 52 49  1 TO 9..160 PRI
8100 4E 54 20 22 20 20 20 20 20 22 2C 49 2C 22 20 2A NT "     ",I," *
8110 22 2C 49 2C 22 20 3D 22 2C 49 2A 49 0D 0A 31 37 ",I," =",I*I..17
8120 30 20 4E 45 58 54 20 49 0D 0A 31 38 30 20 50 52 0 NEXT I..180 PR
8130 49 4E 54 20 5C 20 50 52 49 4E 54 20 22 20 2A 2A INT \ PRINT " **
8140 20 20 20 50 52 4F 47 52 41 4D 4D 20 45 4E 44 45    PROGRAMM ENDE
8150 20 20 20 2A 2A 22 20 5C 20 50 52 49 4E 54 0D 0A    **" \ PRINT..
8160 31 39 30 20 45 4E 44 0D 0A 32 30 30 20 52 45 4D 190 END..200 REM
8170 20 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A  **********
8180 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A **********
8190 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 0D 0A **********..
```

Byte hinter den als Blickfang dienenden 4 Byte „DD“ liegt. Man sieht dort die Byte A081, das bedeutet die Adresse 81A0, und, um 1 vermindert, 819F.
Der CP/M „SAVE“-Befehl erwartet die zu sichernden Daten auf der Adresse 100H, so daß damit auch der Wert von z festliegt. Das Verschieben im Speicher würde also mit dem Befehl m8035, 819F, 100 erfolgen. Zum Sichern des Programmes auf die Platte benötigt der Befehl „SAVE“ noch die Anzahl der „Pages“, der 256 Byte-Blöcke. Dies ist gleich dem Wert der zu sichernden Länge in HEX dividiert durch 100H, +1, falls die Division einen Rest ergibt. Diesen Wert stellt das Programm FORMWAN auf der symbolischen Adresse ANZAHL zur Verfügung. Auf dieser Adresse steht der Wert 016B (vertauscht). Dies ist die Länge des Programmes TEST. Das Programm kann

**Hauptspeicherauszug, Bereich des Programms „FORMWAN“**

```
   *   C S S   *   CHIP SOFTWARE SERVICE                                  PAGE 1
UMWANDELN VON NORTH STAR DOS BASIC DATEIEN IN STANDARD ASCII DATEIEN
1.)   ERLAEUTERUNGEN


                        ;
                        ;*************************************************
                        ;
                        ;
                        ;       DIESES PROGRAMM WANDELT BASIC - QUELLEN-
                        ;       PROGRAMME AUS DEM
                        ;
                        ;        'NORTH STAR - HORIZON' - FORMAT
                        ;
                        ;       UM IN BASIC - QUELLENPROGRAMME IN
                        ;       STANDARD ASCII - ZEICHEN.
                        ;
                        ;       ES WIRD LOGISCH AN DIE STELLE DER
                        ;       DRUCKER ROUTINE DES 'NORTH STAR
                        ;       DOS' - SUPERVISORS EINGEFUEGT
                        ;       UND ERSETZT DIESE.
                        ;
                        ;       DAS PROGRAMM 'DRUCKT' IN EINEN
                        ;       SPEICHERBEREICH, AUS DEM DER ASCII -
                        ;       QUELLEN - CODE DANN IN EINE DATEI
                        ;       AUF EINEM EXTERNEN SPEICHERMEDIUM
                        ;       (KASSETTE ODER DISKETTE) GESICHERT
                        ;       WERDEN KANN.
                        ;
                        ;
                        ;       LETZTE AENDERUNG:   1. JUNI 1981
                        ;
                        ;*************************************************
                        ;
                        ;       EINGABE:        ZEICHEN IN B REG
                        ;       AUSGABE:        ZEICHEN IN A+B REGS
                        ;
                        ;
                        ;-----------------------------------------------
                        ;
                        ;       DEFINITION SYMBOLISCHER NAMEN
                        ;
0000                    PARAMRG ==      B       ;PARAMETER REGISTER
0006                    SPEICHR ==      M       ;SPEICHER ZEIGER
0004                    ARBEITR ==      H       ;ARBEITSREGISTER
                        ;
00FF                    SPVOLL  ==      0FFH    ;'SPEICHER VOLL'-ZEICHEN
                        ;
                        ;
8000                            ORG     8000H   ;ANFANGSADRESSE
                        ;                       DES PROGRAMMES IM
                        ;                       SPEICHER
                        ;
                        ;-----------------------------------------------
                        ;


   *   C S S   *   CHIP SOFTWARE SERVICE                                  PAGE 3
UMWANDELN VON NORTH STAR DOS BASIC DATEIEN IN STANDARD ASCII DATEIEN
+++++ SYMBOL TABLE +++++

ANFANG 8000         ANZAHL 8031         ARBEIT 0004         AUSGAN 8027
BERGR  3F80         INDIKA 802A         PARAMR 0000         SPBER  8035
SPEICH 0006         SPVOLL 00FF         ZAEHLE 802B         ZEIGER 8033
```

also mit dem Befehl SAVE 2 TEST unter dem Namen „Test“ in eine CP/M-Datei gesichert werden.

## Allgemeine Bemerkungen

Zum „Drucken“ eines weiteren BASIC-Programmes muß das Programm FORMWAN erneut in den Speicher geladen werden, um die Zähler zu initialisieren.

Soll ein Programm umgewandelt werden, das größer ist als der verfügbare Speicherbereich, so bricht das Programm FORMWAN den Aufzeichnungsvorgang ab. Ein solches Programm kann dann aber in mehreren Schritten verarbeitet werden.

An syntaktischer Inkompatibilität kann das Programm FORMWAN natürlich nichts ändern. So erkennen andere BASIC-Compiler z.B. nicht das Ausrufungszeichen als Equivalent zum „PRINT“-Befehl. Diese Dinge sind aber zum Teil mit einem Editor-Programm unter CP/M leicht zu ändern.

Der Autor hat auf dem hier erklärten Weg viele Programme verarbeitet und sieht darin eine gute Möglichkeit, die Computer-Unverträglichkeit wenigstens an einer Stelle zu umgehen.

```
   *  QUADRATZAHLEN 1-9:  *

        1 * 1 = 1
        2 * 2 = 4
        3 * 3 = 9
        4 * 4 = 16
        5 * 5 = 25
        6 * 6 = 36
        7 * 7 = 49
        8 * 8 = 64
        9 * 9 = 81

   **    PROGRAMM ENDE    **

READY
*
```

**Beispielausdruck**

```
   *   C S S   *   CHIP SOFTWARE SERVICE                                  PAGE 2
UMWANDELN VON NORTH STAR DOS BASIC DATEIEN IN STANDARD ASCII DATEIEN
2.)   PROGRAMM


   8000                 ANFANG:
   8000    E5                   PUSH    ARBEITR ;REGISTER SICHERN
   8001    3A 802A              LDA     INDIKAT ;SCHALTER LADEN
   8004    B7                   ORA     A       ;SPEICHERBEREICH VOLL?
   8005    C2 8027              JNZ     AUSGANG ;SPRUNG WENN VOLL
   8008    2A 8033              LHLD    ZEIGER  ;SPEICHERZEIGER
   800B    70                   MOV     SPEICHR,B ;ZEICHEN IN SP.-BEREICH
   800C    23                   INX     ARBEITR ;ZEIGER = ZEIGER + 1
   800D    22 8033              SHLD    ZEIGER  ;SICHERN
                        ;
   8010    2A 8031              LHLD    ANZAHL  ;VORWAERTS ZAEHLER
   8013    23                   INX     ARBEITR ;UM 1 ERHOEHEN
   8014    22 8031              SHLD    ANZAHL  ;UND SICHERN
                        ;
   8017    2A 802B              LHLD    ZAEHLER ;SP.-BEREICHSGROESSE
   801A    2B                   DCX     ARBEITR ;ZAEHLER = Z. - 1
   801B    22 802B              SHLD    ZAEHLER ;ZURUECK SPEICHERN
                        ;
   801E    7C                   MOV     A,H     ;DEN ZAEHLER
   801F    B5                   ORA     L       ;AUF NULL PRUEFEN
   8020    2005                 JRNZ    AUSGANG ;SPR WENN NICHT NULL
                        ;
                        ;       SPEICHERBEREICH VOLL, ROUTINE ANHALTEN
                        ;
   8022    3EFF                 MVI     A,SPVOLL ;INDIKATION FUER ...
   8024    32 802A              STA     INDIKAT ;BEREICH VOLL
                        ;
                        ;
   8027    78           AUSGANG:MOV     A,PARAMRG ;ZEICHEN IN A
   8028    E1                   POP     ARBEITR ;WIEDER LADEN
   8029    C9                   RET             ;RUECKSPRUNG INS DOS
                        ;
                        ;
   3F80                 BERGR   ==      3F80H   ;BEREICHER GROESSE
                        ;
   802A    00           INDIKAT:DB      .-.     ;FF = BEREICH VOLL
   802B    3F80         ZAEHLER:DW      BERGR   ;BEREICHS GROESSE
                        ;
   802D    DDDD                 DW      ODDDDH  ;MERKMAL ZUM LEICHTEN
   802F    DDDD                 DW      ODDDDH  ;FINDEN IM SPEICHER
   8031    0000         ANZAHL: DW      .-.     ;VORWAERTS ZAEHLER
   8033    8035         ZEIGER: DW      SPBER   ;BEREICHS ZEIGER
                        ;
   8035                 SPBER:  DS      BERGR   ;SPEICHERBEREICH ZUR
                        ;                       AUFNAHME DES BASIC -
                        ;                       QUELLENPROGRAMMES IN
                        ;                       ASCII - ZEICHEN

                        ;
   8000                         ENDE    ANFANG  ;PROGRAMMENDE
```

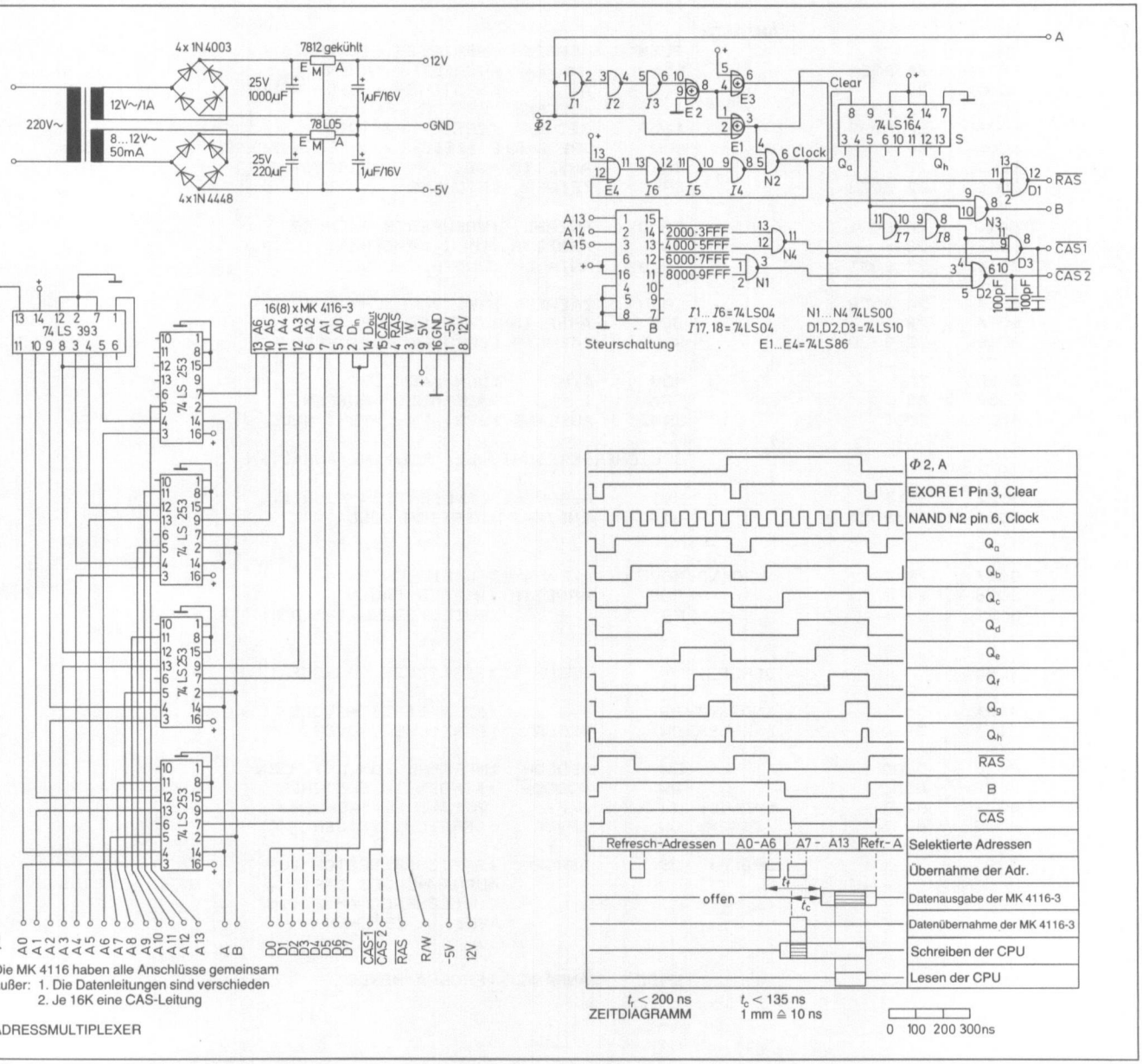
Dieter Nagel

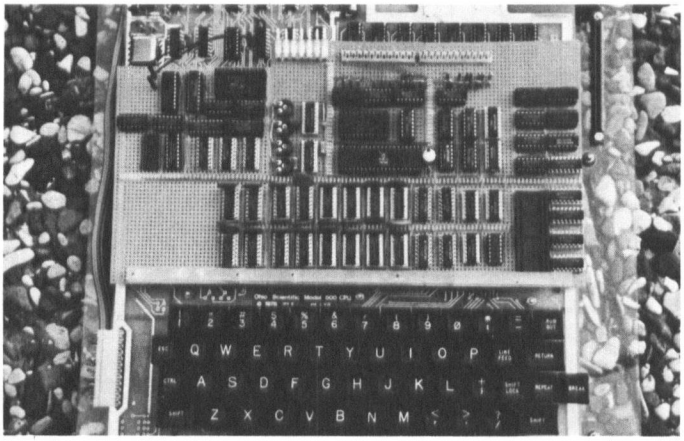
# Dynamische Schreib-Lese-Speicher für 6502-Systeme

Dem Entwurf der folgenden Schaltung ging der Wunsch des Autors voraus, den RAM-Bereich des Superboard II möglichst preiswert zu erweitern. Bei der Kalkulation der Kosten kam der Autor zu der Feststellung, daß die Speichererweiterung mit dynamischen Schreib-Lese-Speicher MK4116-3 (16K × 1 Bit) ca. um den Faktor 4 preiswerter als mit den statischen Speichern 2114-4 (1K × 4 Bit) zu erreichen sein müßte.

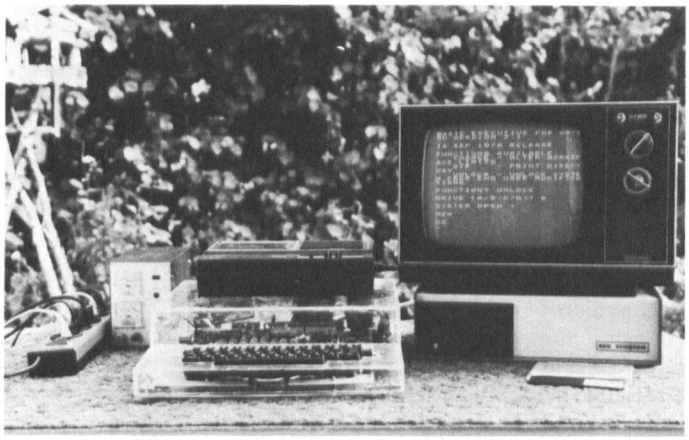
## Gegenüberstellung

Bei der Verwendung von dynamischen Speicher kann sicherlich auf eine Bufferung der Daten- und Adressenleitungen verzichtet werden; der Datenbus wird lediglich mit 2 (1 bei 16K) Eingangs- und 2 (1) Ausgangsleitungen pro Bit verbunden. Die Adressenleitungen werden nur mit den Eingängen der Adressenmultiplexer belastet. Der Aufwand für die Dekodierung der Chip-Select-Eingänge ist aufgrund der „längeren" Organisation geringer. Der Stromverbrauch ist niedriger, speziell die vorhandene 5-V-Spannungsquelle wird kaum belastet. Sie dient lediglich als Referenzspannung für die Eingänge und muß nur die Ströme der Datenausgänge liefern. Zusätzlich werden jedoch die Spannungen +12 V und −5 V benötigt. Die eigentliche Betriebsspannung ist +12 V, sie wird mit 500 (250) mA belastet. In der −5-V-Leitung fließen höchstens 10 mA. Die für die vollständige Adressierung der 4116 not-





**Platine für die DRAM (dynamische Speicher)**



**Gesamtaufbau des Systems vom Autor**

wendigen 14 Adressenleitungen werden nacheinander als sieben Reihen- und sieben Spaltenadressen in die Speicher übernommen. Hierzu ist ein Umschalten der Adressenleitungen notwendig (Multiplex-Betrieb). Außerdem müssen innerhalb von 2 ms alle integrierten Kondensatoren, die die Information speichern, nachgeladen werden. Deshalb ist eine spezielle Steuerschaltung notwendig.

Kurze Beschreibung der zeitlichen Abläufe bei dynamischen Speicher: Der MK4116 besitzt zwei Steuereingänge: $\overline{RAS}$ und $\overline{CAS}$. Diese Eingänge sorgen dafür, daß die Signale an den sieben Adresseneingängen als Reihen- (Row) oder als Zeilenadressen (Column address) gespeichert werden. Dies geschieht bei einer negativen Flanke am jeweiligen Steuereingang. Diese Eingänge steuern außerdem den Zeitpunkt, bei dem die Information am Dateneingang eingelesen wird, und den Zeitpunkt und die Dauer, wann am Datenausgang die Information anliegt. Deshalb müssen die zeitlichen Abläufe der beiden Steuereingänge sehr genau eingehalten werden. Der integrierte Kondensator verliert natürlich durch Leckströme seine Ladung; deshalb muß er nachgeladen werden. Innerhalb von 2 ms muß jede Reihenadresse mindestens einmal angesprochen werden; bei sieben Reihenadressenleitungen ergeben sich $2^7 = 128$ Kombinationen. Während des normalen Betriebs des Computers ist dies nicht sicher gewährleistet. Deshalb muß ein zusätzlicher Nachladezyklus (Refresh cycle) aufgenommen werden.

## Beschreibung der Steuerschaltung

Sämtliche Zeiten werden aus dem Ø2-Signal abgeleitet, es müßte an jedem Expansionsbus zur Verfügung stehen. Während die CPU 6502 mit internen Umwandlungen beschäftigt ist, d.h., während Ø2 = „0", wird der Refresh-Zyklus durchgeführt. Während der aktiven Phase, Ø2 = „1", kann die Kommunikation zwischen der CPU und dem Speicher normal durchgeführt werden. Mit den Invertern I4, I5, I6, dem als Inverter geschalteten EXOR E4 und dem NAND-Gatter N2 ist ein freischwingender Oszillator aufgebaut. Aufgrund der Laufzeiten der Gatter ergibt sich eine Frequenz von 16 MHz. Mit den Invertern I1, I2, I3, dem als Puffer geschalten EXOR E2 und dem EXOR E1 ergeben sich Impulse einer Länge von 40 ns bei jeder Flanke des Ø2-Signals. Diese Impulse synchronisieren den 16-MHz-Oszillator. Gleichzeitig löschen diese Impulse das Schieberegister S. Das 16-MHz-Signal taktet das Schieberegister. Nach n × 60 ns geht der n-te Ausgang ($Q_a$ bis $Q_h$) von „0" nach „1"; an $Q_b$ erfolgt die positive Flanke nach ca. 140 ns und an $Q_c$ nach etwa 200 ns. Das $\overline{RAS}$-Signal ist somit 150 ns nach jedem Wechsel von Ø2 bis zum nächsten Wechsel aktiv (≙ „0"). Die Signale $\overline{CAS\text{-}1}$ und $\overline{CAS\text{-}2}$ sind 230 ns nach der positiven bis zur nächsten negativen Flanke aktiv (≙ „0"); natürlich nur sofern die Adressenbereiche angesprochen werden, d.h. $\overline{Y1}$, $\overline{Y2}$, $\overline{Y3}$ oder $\overline{Y4}$ = „0". Während Ø2 = „0" werden die Refresh-Adressen mittels des Steuersignals A auf die Adresseneingänge der Speicher geschaltet. Wenn Ø2 = „1" ist, werden zuerst die Reihenadressen selektiert (B = „1"). Kurz vor dem $\overline{CAS}$-Signal wird B = „0", d.h., die Adressenmultiplexer schalten jetzt die Spaltenadressen durch. Die Refresh-Adressen werden vom 8-bit-Binärzähler geliefert; somit wird innerhalb von 128 µs jede dekodierte Reihenadresse mindestens einmal angesprochen.

## Problematik der Schaltung

Durch die $\overline{RAS}$- und $\overline{CAS}$-Signale wird der zeitliche Ablauf für die Dateneinund -ausgänge bestimmt. Deshalb ist es notwendig, daß die Verzögerungen vom Ø2-Signal zum $\overline{RAS}$- und $\overline{CAS}$-Signal (150 bzw. 230 ns) genau eingehalten werden. Auch müssen die Refresh- und Reihenadressen vor dem $\overline{RAS}$-Signal und die Spaltenadressen vor dem $\overline{CAS}$-Signal sicher anliegen. Aufgrund von Bauteiletoleranzen, speziell I4, I5, I6, E4 und N2, kann es zu einem Fehlverhalten im zeitlichen Ablauf kommen. Beim Nachbau dieser Schaltung sollte deshalb zuerst die Steuerschaltung aufgebaut werden. Wenn der Oszillator etwas schneller schwingt, kann man sich damit behelfen, daß man das $\overline{RAS}$-Signal von $Q_c$ und das $\overline{CAS}$- und B-Signal von $Q_d$ herleitet. Bei einem zu langsamen Oszillator kann man entsprechend $Q_a$ und $Q_b$ verwenden. Es ist notwendig, daß das $\overline{RAS}$-Signal mindestens 120 ns „1" bleibt. Die Signale sollten mit einem Oszilloskop kontrolliert werden. Wenn die Steuerschaltung funktioniert, können die Adressenmultiplexer und der 8-bit-Binärzähler aufgebaut werden. Man sollte zunächst nur einen MK4116-3 (also nur eine Datenleitung) benutzen und die Schaltung ausprobieren. Sämtliche TTL-Bausteine sollten als LS-Typen verwendet werden. Für ca. je vier TTL-Bausteine (abhängig von der Verdrahtung und der räumlichen Anordnung) sollte ein 100-nF-Kondensator zur Siebung der Versorgungsspannung benutzt werden. Es sollten außerdem die drei Spannungen der RAM-IC mit je einem 100-nF-Kondensator gegen Masse an jedem IC geschaltet werden.

## Weitere Möglichkeiten

Diese Steuerschaltung kann für die CPU 6502A mit einer Taktfrequenz von 2 MHz nicht verwendet werden. Es müßte aber mit einer anderen Steuerschaltung möglich sein, die CPU alle 2 ms anzuhalten und dann alle Reihenadressen anzusprechen. Diese Methode bezeichnet man als Burst-Refresh.

Christian Mentzel

# Fernschreiber T-100 am Apple II

Wer die hohen Ausgaben für einen normalen Drucker scheut, trotzdem aber einen braucht, z.B. für Programmlistings, ist mit einem Fernschreiber gut bedient. Ausrangierte Geräte sind bei Versteigerungen schon ab 60 DM zu haben, überholte Geräte bei den Händlern kosten etwa 500 DM. Das Druckbild ist ausgezeichnet, und es kann neben preiswertem Rollenpapier auch DIN A4 eingespannt werden. Dafür fehlen jedoch einige Zeichen. So sollte man schon beim Kauf darauf achten, ob Groß- oder Kleinbuchstaben verwendet werden. Ebenso wichtig ist die Druckgeschwindigkeit, die bei 50, 75 oder 100 Baud liegen kann. Allerdings erhöht sich mit dem Tempo auch der Lärm, der bei einer 100-Baud-Maschine schon sehr unangenehm werden kann.

```
300LLLL

0300- 8D 8B 03   STA   $038B
0303- 8A         TXA
0304- 48         PHA
0305- 98         TYA
0306- 48         PHA
0307- AD 8B 03   LDA   $038B
030A- C9 8D      CMP   $$8D
030C- D0 0A      BNE   $0318
030E- A9 08      LDA   $$08
0310- 20 47 03   JSR   $0347
0313- A9 02      LDA   $$02
0315- 4C 3A 03   JMP   $033A
0318- 29 3F      AND   $$3F
031A- AA         TAX
031B- BD 94 03   LDA   $0394,X
031E- 8D 8C 03   STA   $038C
0321- 29 20      AND   $$20
0323- CD 8D 03   CMP   $038D
0326- F0 0F      BEQ   $0337
0328- 8D 8D 03   STA   $038D
032B- A8         TAY
032C- F0 04      BEQ   $0332
032E- A9 1B      LDA   $$1B
0330- D0 02      BNE   $0334
0332- A9 1F      LDA   $$1F
0334- 20 47 03   JSR   $0347
0337- AD 8C 03   LDA   $038C
033A- 20 47 03   JSR   $0347
033D- 68         PLA
033E- A8         TAY
033F- 68         PLA
0340- AA         TAX
0341- AD 8B 03   LDA   $038B
0344- 4C F0 FD   JMP   $FDF0
0347- 8D 58 C0   STA   $C058
034A- 20 6B 03   JSR   $036B
034D- A0 04      LDY   $$04
034F- 4A         LSR
0350- 90 09      BCC   $035B
0352- 8D 59 C0   STA   $C059
0355- 20 6B 03   JSR   $036B
0358- 4C 61 03   JMP   $0361
035B- 8D 58 C0   STA   $C058
035E- 20 6B 03   JSR   $036B
0361- 88         DEY
0362- 10 EB      BPL   $034F
0364- 8D 59 C0   STA   $C059
0367- 20 7B 03   JSR   $037B
036A- 60         RTS
036B- A2 0E      LDX   $$0E
036D- 8E 8E 03   STX   $038E
0370- A2 C0      LDX   $$C0
0372- CA         DEX
0373- D0 FD      BNE   $0372
0375- CE 8E 03   DEC   $038E
0378- D0 F6      BNE   $0370
037A- 60         RTS
037B- A2 15      LDX   $$15
037D- 8E 8E 03   STX   $038E
0380- A2 C0      LDX   $$C0
0382- CA         DEX
0383- D0 FD      BNE   $0382
0385- CE 8E 03   DEC   $038E
0388- D0 F6      BNE   $0380
038A- 60         RTS
```

**Fernschreibprogramm für den Apple II**

## Der Anschluß

Hat man nun mit List und Tücke einen Fernschreiber ergattert und schließt ihn probenhalber ans Netz an, wird er mit einem Heidenlärm loslaufen, ohne einen Buchstaben zu drucken. Das liegt daran, daß er zum Betrieb eine „Linienstromquelle" benötigt. Tastatur und Druckwerk sind nämlich mechanisch unabhängig voneinander.

Schaltbild zeigt die entsprechende Schaltung. Die hohe Spannung von 80 Volt ist dabei notwendig, um an dem Empfangsmagneten einen möglichst gleichmäßigen Stromverlauf zu erzeugen. Im Anschlußkabel des Fernschreibers finden Sie vier Drähte vor. Zwei davon (mit einem Widerstand von etwa 80 Ω) gehören zum Empfangsmagnet, die beiden anderen zum Kontaktgeber der Tastatur.

Wenn Sie nun Kontaktgeber und Empfangsmagnet in Reihe an die Stromversorgung legen (Vorsicht Hochspannung!) und den Strom mit P1 auf 40 mA einstellen, hört das Druckwerk auf die Tastatur.

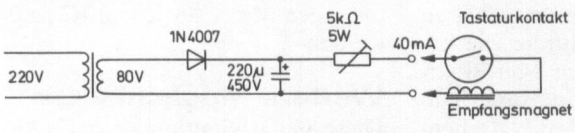
In der Praxis allerdings können zwei Fehler auftreten. Entweder das Druckwerk reagiert überhaupt nicht, was auf fehlerhafte Verbindungen, ein Nichtfunktionieren des Tastaturkontakts oder einen fehlerhaften Sitz des Empfangsmagneten hindeuten kann. Oder Fehlschriften, die man an einem skalierten Regler korrigieren kann. Beim Siemens T-100 liegt dieser Regler vorne rechts vor dem Empfangsmagneten.

Sind diese Schwierigkeiten überwunden, können Sie das Interface anschließen. Wegen der hohen Spannung ist ein Optokoppler sehr zu empfehlen. Sicherheitshalber ist das Interface nochmals zweigeteilt, wobei Teil a) im Computer unterzubringen ist. Der Ausgang ist kurzschlußfest.

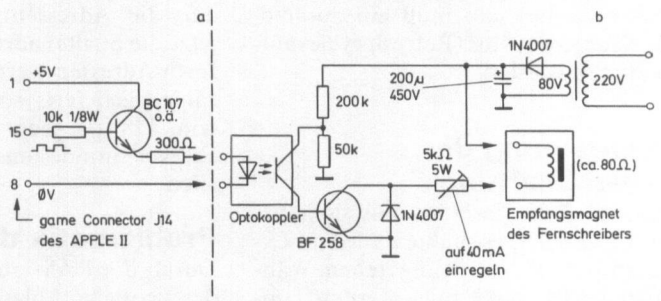
Damit wäre die lästige Hardware erledigt, und wir können uns der Freude aller „Computerfreaks" zuwenden: der Software und, der Geschwindigkeit zuliebe, gleich in Maschinensprache.

## Die Software

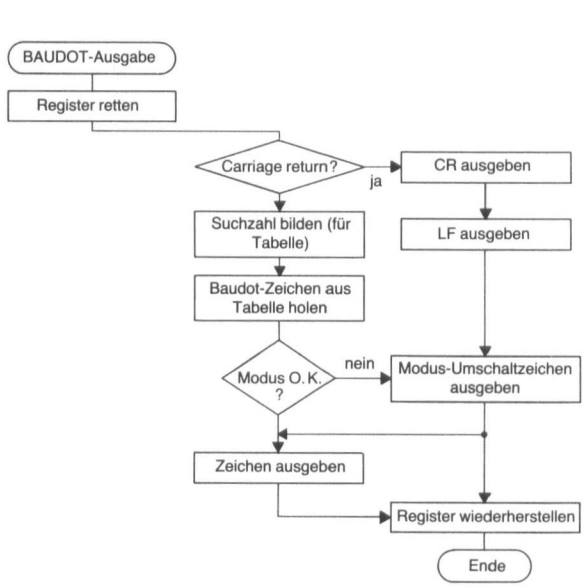
Unser Problem liegt darin, den parallelen ASCII-Kode in den seriellen Baudot-Kode zu übertragen, also in eine bestimmte Folge von An-Aus-Umschaltun-



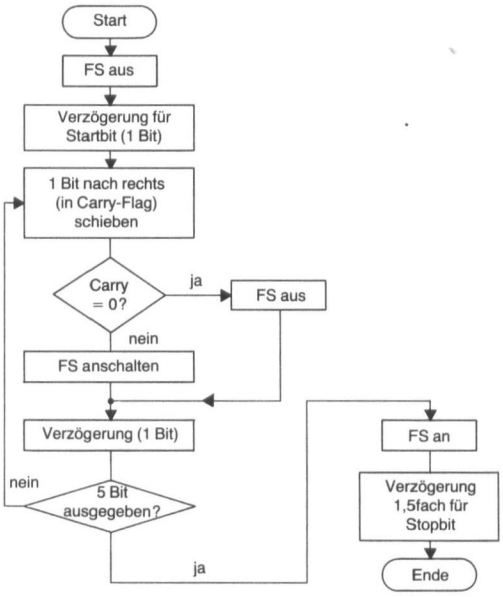

**Schaltbild 1: Linienstromversorgung**




**Schaltbild 2: Fernschreiber-Interface**




**Programm BAUDOT-Ausgabe**




**Subroutine Zeichenausgabe**

gen mit einer Impulsdauer von 20ms (bei 50 Baud – bei 100 Baud wären es 10ms). Der Baudot-Kode besteht aus fünf datenübertragenden Bit, geführt von einem Startbit (aus) und gefolgt von einem Stoppbit (an) mit der 1,5fachen Impulslänge. Eine zusätzliche Schwierigkeit besteht in der Unterscheidung von Buchstaben- und Zeichenmodus, die etwa mit der Funktion einer „Shift-lock"-Taste zu vergleichen sind. Um zwischen beiden Modi umschalten zu können, wird je ein „Umschaltzeichen" benötigt.

Das nebenstehende Programm paßt in den nicht verwendeten Teil des Apple-Speichers. Es wird jedesmal angesprungen, bevor der Apple ein Zeichen auf den Bildschirm druckt. Dabei befindet sich das zu druckende Zeichen im Akkumulator. Die Einschaltung des Druckprogramms in den Ausgabestrom übernimmt Programm 2, das auf Monitorebene mit CTRL-Y angesprochen werden kann bzw. in BASIC und Applesoft mit CALL 1016. Die Ausschaltung des Druckers erfolgt mit RESET bzw. PR = 0.

Das Programm rettet zunächst alle Register, prüft dann das zu druckende Zeichen auf einen Wagenrücklauf und gibt in diesem Fall CR und LF (line-feed) aus. Handelt es sich um ein anderes Zeichen, werden die ersten beiden Bit gelöscht und mit der so entstandenen Suchzahl das zu druckende Zeichen aus der Tabelle gesucht. Da nicht alle ASCII-Zeichen im Baudot-Alphabet vorkommen, werden manche als Leerzeichen bzw. ein anderes Zeichen gedruckt. Das aus der Tabelle geholte Zeichen enthält in Bit 5 den Modusvermerk. Ist dieser Modus anders als beim zuletzt gedruckten Zeichen, wird zunächst die Modus-Umschaltung ausgegeben. Nach der Ausgabe des Zeichens werden die Register wiederhergestellt und die Bildschirmausgabe angesprungen.

Die eigentliche Druckroutine (ab Ø347) arbeitet mit Output-Port 1 des „game-connectors". Mit CØ59 wird dieser angeschaltet, mit CØ58 ausgeschaltet. Wichtig ist dabei allein das Ansprechen der Adresse. Nach Ausgabe des Startbit werden die datenübertragenden Bit nacheinander rechts aus dem Akkumulator geschoben und der Output-Port je nach Inhalt des Carry-Flags an- oder ausgeschaltet. Das Y-Register zählt die fünf auszugebenden Bit ab. Zum Schluß wird das Stoppbit (1,5fache Verzögerung) ausgegeben.

Die Warteschleifen (ab Ø36B bzw. Ø37B) sind auf die Betriebsfrequenz des Apple und einen 75-Baud-(= 100 words/min) Fernschreiber geeicht, für einen 100-Baud-Fernschreiber sind die Zellen Ø371 und Ø381 von CØ auf 9Ø zu ändern.

Dem in Maschinensprache Unerfahrenen sei zu raten, sich das Apple-Manual anzusehen, wo das Eingeben eines Maschinenprogramms detailliert beschrieben wird. Auch die Anschlußbelegungen des game-connector sind dort zu finden.

Denken Sie aber bei aller Bastlerfreude daran, den Apple auszuschalten, bevor Sie daran herumstöpseln und alle Leitungen lieber dreimal nachzuprüfen, denn einen „VOLTAGE-ERROR" wird er Ihnen bestimmt nicht anzeigen.

Und nun viel Spaß mit dem vielleicht billigsten Drucker der Welt!

```
3E0L
03E0-   A5 37     LDA   $37
03E2-   8D 46 03  STA   $0346
03E5-   A5 36     LDA   $36
03E7-   8D 45 03  STA   $0345
03EA-   A9 03     LDA   $$03
03EC-   85 37     STA   $37
03EE-   A9 00     LDA   $$00
03F0-   85 36     STA   $36
03F2-   60        RTS
03F3-   EA        NOP
03F4-   EA        NOP
03F5-   EA        NOP
03F6-   EA        NOP
03F7-   EA        NOP
03F8-   4C E0 03  JMP   $03E0
03FB-   EA        NOP
03FC-   EA        NOP
03FD-   EA        NOP
03FE-   EA        NOP
03FF-   EA        NOP
▽
```

**FS-Initialisierung**

```
▽
▽394.3D3

0394- 00 03 19 0E
0398- 09 01 0D 1A 14 06 0B 0F
03A0- 12 1C 0C 18 16 17 0A 05
03A8- 10 07 1E 13 1D 15 11 2D
03B0- 00 34 2B 00 24 2D 31 29
03B8- 29 3D 3A 2B 2F 32 34 23
03C0- 2C 23 3C 3D 36 37 33 21
03C8- 2A 30 35 27 26 38 2E 3E
03D0- 24 2E 24 39
▽
```

**ASCII-Baudot-Tabelle**

Gustav Wostrack

# Videoerweiterung für den NASCOM-1

Diese Videoerweiterung für den NASCOM-1 macht sich das nutzlose achte Bit des Video-RAM zunutze und erlaubt folgende optische Effekte:

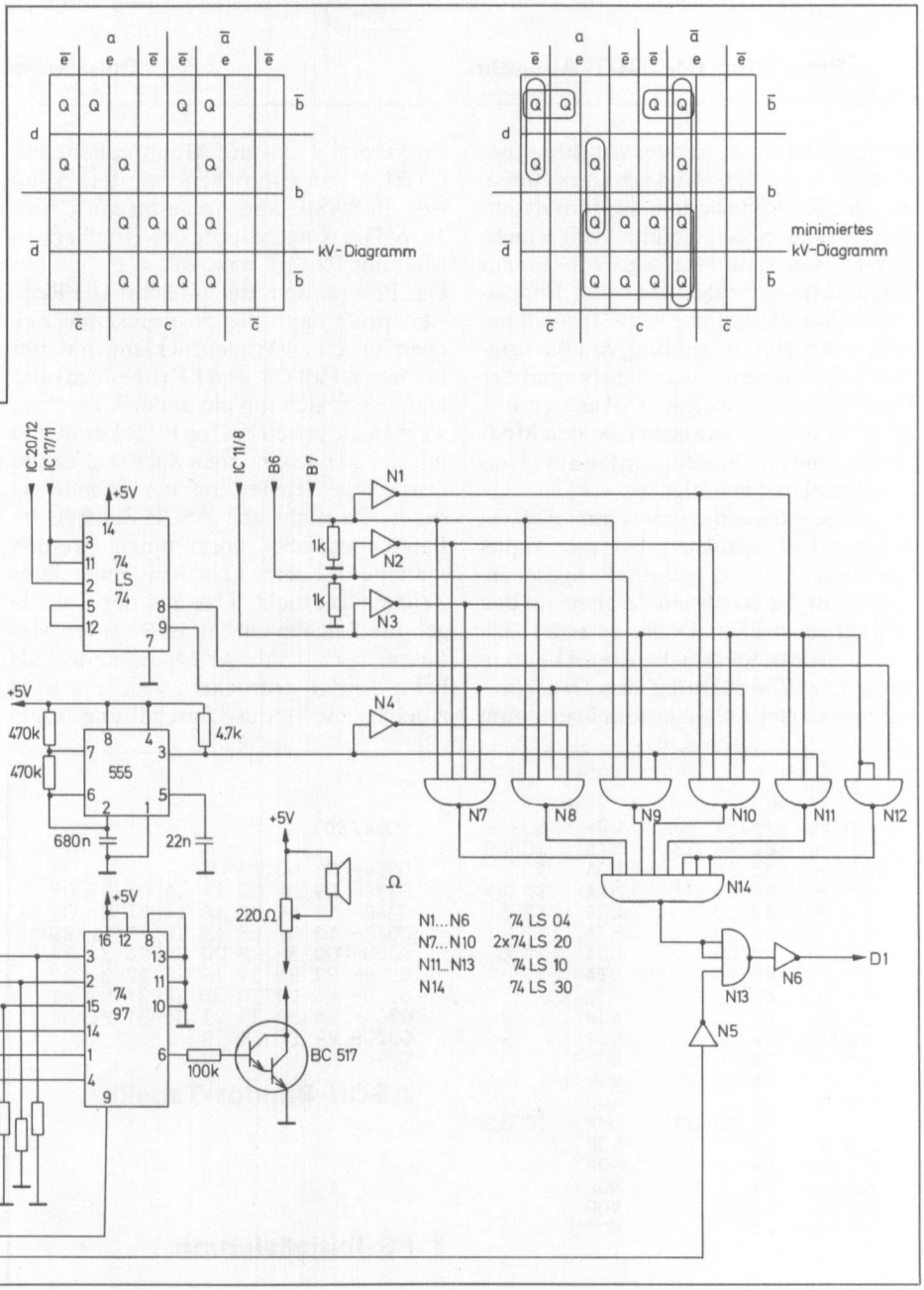
- inverse Darstellung von einzelnen frei wählbaren Zeichen,
- blinkende Darstellung von einzelnen Zeichen in zwei Versionen,
  a) Wechsel zwischen Zeichen und blank,
  b) Wechsel zwischen Zeichen und inversem Zeichen.

## Beschreibung und Anwendung

Darüber hinaus wurde die Schaltung so konzipiert, daß ein neuer Zeichengenerator eingesetzt werden kann, der dann speziell auf die Belange des Hobbyisten zugeschnitten ist. Unter den 128 Zeichen wären dann beispielsweise die Spielkartenfarben vertreten wie auch die Signaturen für die Schachfiguren. Verbunden mit der Möglichkeit, inverse und blinkende Zeichen darstellen zu können, ergeben sich sehr realistische Effekte, die sonst nur mit hohem Aufwand in teuren Graphiksystemen realisierbar sind.

Dieser Zeichengenerator ist als EPROM 2708 ausgeführt. Auf der Platine der Videoschaltung sind zwei Steckplätze (24polig) vorgesehen, die so beschaltet sind, daß das 2708 einmal für Systeme mit auch ursprünglichem 2708-Zeichengenerator paßt, aber auch für solche, die von Hause aus mit MCM 6576 o.ä. (wie beispielsweise NASCOM-1) ausgerüstet sind. Dies ist für diejenigen interessant, die diese Erweiterung nicht in den NASCOM-1, sondern in einen anderen $\mu$-Computer einbauen wollen.

Außerdem wurde ein programmierbarer Teiler installiert, der über einen eingebauten Lautsprecher die entsprechende



akustische Untermalung bei Videospielen übernimmt. Alle beschriebenen Effekte können per Software abgerufen werden. Dazu werden die Anschlüsse $B_0$ bis $B_7$ an einen 8-bit-Port angeschlossen.
Die Anschlüsse $B_0$ bis $B_7$ haben dabei folgende Bedeutung:

| $B_7$ | $B_6$ | |
|---|---|---|
| 0 | 0 | Grundzustand, Bit 7 hat keinen Einfluß |
| 0 | 1 | ausgewählte Zeichen blinken (Wechsel zwischen Zeichen und blank) |
| 1 | 0 | ausgewählte Zeichen sind invers dargestellt |
| 1 | 1 | ausgewählte Zeichen blinken (Wechsel zwischen Zeichen und inversem Zeichen) |

Unter ausgewählten Zeichen sind alle diejenigen zu verstehen, die das Format $1XXX\ XXXX_2$ haben.

$Bi - B_5$ Teilerverhältnis des programmierbaren Teilers von 00000 bis $11111_2$, d.h. von 0 bis $63_{10}$; bei einer Eingangsfrequenz von 7812 Hz ergeben sich am Ausgang Frequenzen von 7812 bis 124 Hz.

Beispiel:
Die Kombination $9E_{16}$ erzeugt folgenden Effekt:

a) alle Zeichen, die im Video-RAM das Format $1XXX\ XXX_2$ haben, werden invers dargestellt,
b) am Teilerausgang steht eine Frequenz von 7812 Hz/30 = 260 Hz an.

## Entwurf der Schaltung

Zunächst wurde das Konzept festgelegt, also die Frage geklärt, was die Schaltung eigentlich leisten soll. Nachdem diese Merkmale formuliert waren, wurden die Probleme schrittweise gelöst.
Zunächst sollte die Möglichkeit geschaffen werden, einen neuen Zeichengenerator einzusetzen, der auch selbst entworfen und realisiert, d.h. programmiert werden konnte. Dabei fiel die Wahl auf das EPROM 2708. Also mußte dafür ein neuer Steckplatz geschaffen werden. Allerdings sollte die Möglichkeit, den alten Zeichengenerator zunächst weiterzubenutzen, erhalten bleiben. Die Forderung für den neuen Steckplatz wurde so erfüllt, daß einfach alle Anschlüsse des alten MCM6576 zu den entsprechenden Pins des neuen EPROM 2708 geführt wurden. Die Verbindungen zwischen der Hauptplatine und der Erweiterung wurden durch eine Wire-Wrap-Fassung hergestellt, die einmal die Erweiterungsplatine trägt, – d.h., die Wire-Wrap-Stifte sind auf der der Fassung zugewandten Seite mit der Platine verlötet – und andererseits sind diese Stifte in die Fassung des ursprünglichen Zeichengenerators auf der Hauptplatine gesteckt. Von hier aus werden dann die Verbindungen zum neuen Steckplatz für das 2708 gemäß Schaltplan hergestellt.
Als akustische Untermalung für Videospiele sollte eine Schaltung entworfen werden, die Töne verschiedener Frequenz erzeugt und hörbar macht, ohne aber die Spielprogramme softwaremäßig zu belasten und zu verkomplizieren. Die Grundfrequenz hierzu wird aus den Ansteuersignalen gewonnen, die der Zeichengenerator als Adressen erhält. Es wurde die Frequenz 7812 Hz ausgewählt. Um den Softwareaufwand bei der Erzeugung der verschiedenen Frequenzen gering zu halten, wurde ein programmierbarer Teiler eingesetzt, nämlich das IC 7497. Es bietet die Möglichkeit, das Teilerverhältnis von 0 bis 63 mit der Schrittweite 1 einzustellen. Dazu werden sechs Leitungen benötigt, die die binär kodierte Information für das Teilerverhältnis an das IC übermitteln. Diese sechs Leitungen werden an einen 8-bit-Port angeschlossen. Der gesamte Softwareaufwand beschränkt sich also auf das Laden des Interface. Das Ausgangssignal des IC wird durch einen Darlingtontransistor verstärkt und kann dann dem Lautsprecher zugeführt werden.
Das nutzlose achte Bit im Video-RAM sollte nutzbar gemacht werden. Zusammen mit einem weiteren Kode, der einer zu entwerfenden Schaltung zuzuführen ist, sollte dieses Bit das Entscheidungskriterium dafür sein, ob der durch den Kode festgelegte Videoeffekt eintritt oder nicht. In diesem Zusammenhang wurde folgendes festgelegt:

- Der Kode soll zweistellig sein;
- der Effekt soll eintreten, wenn das achte Bit logisch „1" ist;
- folgende Effekte sollen dem Kode zugeordnet sein:

| Kode | |
|---|---|
| Effekt | |
| 0 0 | Grundzustand, normales Videobild mit weißen Zeichen auf schwarzem Hintergrund |
| 0 1 | ausgewählte Zeichen blinken (Wechsel Zeichen-blank) |
| 1 0 | ausgewählte Zeichen sind invers dargestellt |
| 1 1 | ausgewählte Zeichen blinken (Wechsel Zeichen – inv. Zeichen) |

Die Forderungen wurden in einem KVD dargestellt und minimiert.
Es ergeben sich folgende Bedingungen

a... Kode mit Stellenwert $1_7$ $B_7$
b... Kode mit Stellenwert 0 $B_6$
c... Videosignal vom Schieberegister IC 11/8
d... Bit 7 des Video-RAM IC 20/12
e... Blinksignal
Q... Ausgangssignal zur Zusatzlogik bzw. Diode D1

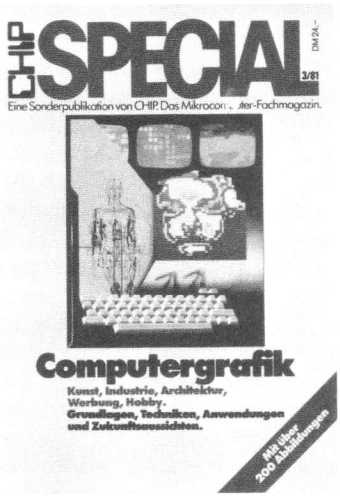
Daraus ergibt sich folgende Gleichung
$Q = a\bar{b}cd \vee a\bar{c}d\bar{e} \vee d\bar{d} \vee \bar{a}ce \vee abce \vee \bar{a}\bar{b}cd$.
Um diese mit NAND-Gliedern realisieren zu können, wird das Gesetz von De Morgan angewandt:
$Q = a\bar{b}cd \wedge a\bar{c}d\bar{e} \wedge c\bar{d} \wedge \bar{a}ce \wedge abce \wedge \bar{a}\bar{b}cd$.
Allein aus dieser Gleichung ergibt sich die Verschaltung der Gatter N1 – N4, N7 – N12 und N14.
Allerdings kann das Bit 7 nicht direkt an diese Schaltung angeschlossen werden, da es sonst zeitlich versetzt, d.h. zu früh an diese Logik gelangt. Die Erklärung hierfür ist ganz einfach. Die Bit 0 bis 6 des Video-RAM durchlaufen noch die Bausteine IC17 und IC16 (Auffangflipflops und Zeichengenerator), bis sie in das Schieberegister IC15 gelangen. Diese Laufzeit gilt es auszugleichen. Dazu werden zwei D-Flipflops benutzt, die genauso wie IC17 (Auffangflipflops) getaktet werden (Anschluß IC 17/11).
Für das Blinksignal ist ein Rechtecksignal mit einer sehr niedrigen Frequenz notwendig. Da diese im NASCOM-1 nicht erzeugt wird, ist es unumgänglich, hierfür eigens einen Baustein NE555 einzusetzen. Durch Variation der Komponenten kann dessen Frequenz geändert werden.
Die Umschaltlogik mit N5, N6 und N13 dient dazu, bei invertiertem Betrieb die untere Hälfte des Zeichenfeldes (ein Zeichenfeld ist doppelt so hoch wie ein Zeichen) auszublenden bzw. zu verdunkeln.
Die beiden Eingänge, die im KV-Diagramm als a und b bezeichnet sind, werden zusammen mit den sechs Anschlüssen für den programmierbaren Teiler an ein 8-bit-Port angeschlossen. Alles Weitere kann dem Schaltplan entnommen werden. Die Anschlüsse beziehen sich auf den NASCOM-1.

## Die notwendige Software

Wie schon aus den gestellten Forderungen hervorging, soll der Softwareaufwand gering gehalten werden. Er beschränkt sich einmal auf die Programmierung eines 8-bit-Ports und der Speicherstellen im Video-RAM, die von dem gewünschten Effekt betroffen sein sollen.

# Premiere im Computeralltag

**Es geht um Kunst. Kunst aus der Maschine. Genauer gesagt: Es geht um Computerkunst.**
CHIP, das erfolgreiche Mikrocomputer-Fachmagazin, bringt erstmalig in seiner vielbeachteten Sonderreihe CHIP SPECIAL eine Ausgabe, die sich ausschließlich mit **Computergrafik** beschäftigt. Nirgendwo anders können Sie in einem Stück so komplex alles über die verschiedensten Anwendungsbereiche der grafischen Datenverarbeitung erfahren. Top-Gelegenheit für all jene, die in Kunst, Architektur, Industrie, Werbung und Hobby mit dem Mikrocomputer umgehen oder umgehen wollen. CHIP lüftet das Geheimnis: Wie eindrucksvolle Computergrafiken entstehen. Was auf dem Drucker, Plotter oder Bildschirm schon heute möglich ist. Was sich zukünftig abzeichnet. Und, und. Einfach aufregend! Lesen Sie alles über Computergrafik in **Computergrafik**, der neuesten Sonderpublikation der attraktiven CHIP SPECIAL-Serie. Denn **Computergrafik** muß man einfach haben. Jetzt besorgen.




**CHIP** und CHIP SPECIALS gibt es im Zeitschriften-, Buch- und Elektronikfachhandel. Oder einfach bestellen beim **CHIP-Leserservice,** Vogel-Verlag, Postfach 67 40, D-8700 Würzburg 1.

## Computergrafik.

☐ Ja. Senden Sie mir das neueste CHIP SPECIAL 3/81 **Computergrafik.** Preis: 24,– DM.* Ausl.: 30,– DM.*
☐ Ja. Senden Sie mir folgende CHIP SPECIALS zum Preis von 24,– DM* je Expl.,
Ausl.: 30,– DM.*
____ Expl. Special 1/81: **Mikrocomputer-Hardware-Katalog**
____ Expl. Special 2/81: **Programme II**
____ Expl. Special 1/80: **Mikrocomputer-Digest**
____ Expl. Special 2/80: **Programme I**
____ Expl. Special 4/80: **Mikroprozessoren**
Zutreffendes ist angekreuzt!

Name/Vorname

Straße

PLZ/Ort

Datum/Unterschrift

*Zuzüglich 3,50 DM Versandkostenanteil. Inlandspreise einschl. MwSt.



CHIP SPECIAL

Bauanleitung für Mikrocomputer

1981

ISBN 3-8023-0702-X

**Redaktionsdirektor:** Richard Kerler (verantwortlich für den Inhalt)
**Redaktion:** Ruth Bosch, Herbert Bernstein
**Chef vom Dienst:** Axel Schneider
**Gestaltung:** Ludwig Lechler
**Titelillustration:** Ernst Jünger
**Redaktionsservice:** Marianne Weissbach
**Redaktion:** Vogel-Verlag KG Würzburg, Redaktion CHIP, Bavariaring 9, 8000 München 2, Telefon (0 89) 53 03 02, Telex 05 216 449, Telefax (0 89) 535 000
**Verlag:** Vogel-Verlag KG, Postfach 67 40, D-8700 Würzburg 1, Tel. (09 31) 41 02-1, Telex 06 8 883, Telefax (09 31) 41 02-5 29, Telegramme: CHIP-Würzburg
**Verlagsleiter:** Hugo E. Martin, Würzburg
**Anzeigenleiter:** Harald Kempf, Würzburg (verantwortlich für Anzeigen)
**Anzeigenservice:** Margit Ebert, Frauke Reith: Postfach 67 40, 8700 Würzburg 1, Telefon (09 31) 41 02-3 59, Telex 06 8 883
**Vertriebsleitung:** Axel Herbschleb, Würzburg; Generalvertretung Österreich: Fachbuch Center Erb, Amerlingstr. 1, A-1061 Wien 6, Telefon (02 22) 56 62 09; Generalvertretung Schweiz: Thali AG, CH-6285 Hitzkirch, Telefon (0 41) 85 28 28.
**Vertrieb Handelsauflage:** Vereinigte Motor-Verlage GmbH, Leuschnerstr. 1, 7000 Stuttgart 1, Tel. (07 11) 20 43-1, Telex 07 22 036
**Bezugsmöglichkeiten:** Bestellungen nehmen der Verlag, die o.a. Generalvertretungen, jedes Postamt und alle Buchhandlungen im In- und Ausland entgegen. Sollte die Zeitschrift aus Gründen, die nicht vom Verlag zu vertreten sind, nicht geliefert werden können, besteht kein Anspruch auf Nachlieferung oder Erstattung vorausbezahlter Bezugsgelder.
**Bankverbindungen:** Bayerische Vereinsbank, Würzburg, Dresdner Bank AG, Würzburg, Postscheckkonto: Vogel-Verlag, Postscheckamt Nürnberg.
**Gesamtherstellung und Versand:** VOGEL-DRUCK WÜRZBURG, Max-Planck-Straße 7/9, D-8700 Würzburg.



Unverlangte Manuskripte werden nur zugesandt, wenn Rückporto beigefügt ist. — Für die mit Namen oder Signatur des Verfassers gekennzeichneten Beiträge übernimmt die Redaktion lediglich die presserechtliche Verantwortung. — Die in dieser Zeitschrift veröffentlichten Beiträge sind urheberrechtlich geschützt. Übersetzung, Nachdruck, Vervielfältigung sowie Speicherung in Datenverarbeitungsanlagen nur mit ausdrücklicher Genehmigung des Verlages. Jede im Bereich eines gewerblichen Unternehmens hergestellte oder benutzte Kopie dient gewerblichen Zwecken gem. § 54 (2) UrhG und verpflichtet zur Gebührenzahlung an die VG Wort, Abteilung Wissenschaft, Goethestraße 49, 8000 München 2, von der die Zahlungsmodalitäten zu erfragen sind.
Die Redaktion hat die Manuskripte, Bauanleitungen und Schaltungen sorgfältig geprüft. Für Fehler im Text, in Schaltbildern, Aufbauskizzen, Stücklisten usw., die zum Nichtfunktionieren oder evtl. zum Schadhaftwerden von Bauelementen führen, kann keine Haftung übernommen werden.
Sämtliche Veröffentlichungen in CHIP erfolgen ohne Berücksichtigung eines eventuellen Patentschutzes, auch werden Warennamen ohne Gewährleistung einer freien Verwendung benützt.

# CHIP WISSEN

Die Buchreihe von CHIP, mit der Sie Ihr Mikrocomputer-Wissen systematisch vertiefen können. Für Einsteiger, Fortgeschrittene und Profis. Band für Band geballte Fachinformation.

Baumann, Rüdeger
**Programmieren mit PASCAL**
260 Seiten, 20 DM,
ISBN 3-8023-0667-8

Chirlian, Paul M.
**PASCAL**
(in englischer Sprache), 212 Seiten,
30 DM, ISBN 3-8023-0666-X

Didday, Rich
**Mein Computer – $2^{10}$ Fragen und Antworten**
Band 1: Hardware, 298 Seiten,
36 DM, ISBN 3-8023-0659-7
Band 2: Software, 240 Seiten,
38 DM, ISBN 3-8023-0660-0

Sacht, Hans Joachim
**$\mu$P-Programmierfibel**
352 Seiten, 38 DM,
ISBN 3-8023-0644-9

Willis, Jerry/Pol, Bernd
**Was der Mikrocomputer alles kann**
366 Seiten, 33 DM,
ISBN 3-8023-0643-0

Zaks, Rodnay
**Programmierung des 6502**
328 Seiten, 44 DM,
ISBN 3-8023-0686-4

Zaks/Lesea
**Mikroprozessor-Interface-Techniken**
446 Seiten, 44 DM,
ISBN 3-8023-0685-6

**CP/M Handbuch mit MP/M**

Zaks, Rodnay

340 Seiten, 44 DM
ISBN 3-8023-0704-6

**Mikrocomputer-Betriebssysteme**

Schmidt, Kl./Renner, G.

CP/M, CDOS, DOS
152 Seiten, 25 DM
ISBN 3-8023-0655-4

**Wie man in BASIC programmiert**

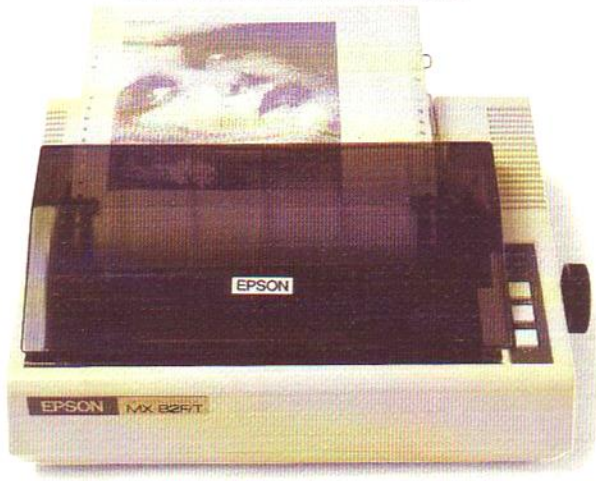
Pol, Bernd

368 Seiten, 30 DM
ISBN 3-8023-0637-6

**Alle Bücher erhalten Sie bei Ihrem Buchhändler**

# DER NEUE EPSON MX-82 F/T. ER KANN BEIDES: DRUCKEN WIE DER MX-80 F/T. UND AUCH PLOTTEN.

DER NEUE EPSON MX-82 F/T.




Wir übernehmen keine Garantie bei Einbau nicht von uns autorisierter Teile oder Nachrüstsätze!

Fordern Sie unsere Händlerliste an.

Plotten (High Resolution-Grafic). Normal Mode: 8 × 576 dots/Zeile.
Dual Density Mode: 8 × 1152 dots/Zeile. Zeichen/Zeile: 40, 66, 80, 132/6.7". 48, 79, 96, 159/8.0".
Interface: Apple II (8132 + 8231).

Funktionsvergleich:

| | MX-82 F/T | MX-80 F/T | | MX-82 F/T | MX-80 F/T | | MX-82 F/T | MX-80 F/T |
|---|---|---|---|---|---|---|---|---|
| CR | + | + | DC3 | + | + | ESC E | + | + |
| LF | + | + | BEL | + | + | ESC G | + | + |
| FF | + | + | BS | + | – | ESC K | + | – |
| SO | + | + | FSCO | + | + | ESC L | + | – |
| SI | + | + | FSC2 | + | + | ESC N | + | – |
| HT | + | + | FSC8 | + | + | ESC Q | + | – |
| VT | + | + | FSC A | + | + | ESC R | + | – |
| DC1 | + | + | FSC C | + | + | | | |

+ = ja – = nein

EPSON

Die Zeit ist da. Für bessere Ideen.

Kostenlos und unverbindlich informiert Sie:
EPSON Deutschland GmbH, Am Seestern 24, 4000 Düsseldorf. Telefon (02 11) 59 61 001. Techn. Abt. (02 11) 59 61 007. Telex 8 584 786.
EPSON ist der führende Hersteller von Mini-Printern. Weltweit. Mit einem Marktanteil von ca. 70%. Ein Tochterunternehmen der SEIKO-Group.